National Library of Canada
Bibliothèque nationale du Canada

Canadian Theses Service
Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, tests publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30.

NL-339 (r. 88/04)

Canada

THE UNIVERSITY OF ALBERTA

DESIGN STUDY OF A ROBOTIC ARM FOR USE BY QUADRIPLEGICS

BY

RANDALL KIMURA

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL ENGINEERING

EDMONTON, ALBERTA

FALL 1987

Permission has been granted to the National Library of Canada to microfilm this thesis and to lend or sell copies of the film.

The author (copyright owner) has reserved other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without his/her written permission.

L'autorisation a été accordée à la Bibliothèque nationale du Canada de microfilmer cette thèse et de prêter ou de vendre des exemplaires du film.

L'auteur (titulaire du droit d'auteur) se réserve les autres droits de publication; ni la thèse ni de longs extraits de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation écrite.

ISBN 0-315-40953-3

THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: RANDALL KIMURA

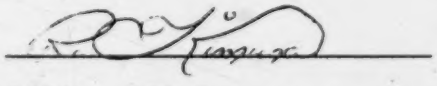
TITLE OF THESIS: DESIGN STUDY OF A ROBOTIC ARM FOR USE BY QUADRIPLEGICS

DEGREE: MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

YEAR THIS DEGREE GRANTED: 1987

Permission is granted to THE UNIVERSITY OF ALBERTA LIBRARY to reproduce single copies of this thesis and to lend or sell such copies for private, scholary or scientific research purposes only.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.



7623 - 152 B Avenue

Edmonton, Alberta

Date: Sept 29/87

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled Design Study of a Robotic arm for use by Quadriplegics submitted by Randall Kimura in partial fulfilment of the requirements for the degree of Master of Science in Electrical Engineering.

(Supervisor)

Roger W Toogood

Terry Hyde Blois

Date: Apr 24-87

ABSTRACT

The robotic arm, fitted with an eating utensil, may be used by a quadriplegic with a minimum amount of aid. The versatility of this device may be increased with the addition of a remote control interface which will allow the user to perform various everyday tasks. When audio input is employed, the largest number of handicapped persons may benefit, independent of the degree or type of handicap. For patients who do not possess the required vocal capabilities, a joystick or keypad interface may be adapted to the software. Ideally, the cost expenditure must be kept to a minimum in order to allow the largest number of users access.

TABLE OF CONTENTS

1. INTRODUCTION.....1

2. DESIGN CONSIDERATIONS.....6
2.1. STEPPER MOTORS.....6
2.1.1. ZERO CROSSING DETECTOR.....8
2.1.2. LEVEL SHIFTER.....11
2.1.3. PHASE SEQUENCE GENERATOR.....13
2.1.4. DIRECTION DETECTION.....17
2.1.5. SPEED CONTROL.....23
2.2. COORDINATE CALCULATIONS.....24

3. HARDWARE.....33
3.1. ACIA.....33
3.2. FLOATING POINT ROM.....34
3.3. MEMORY.....36
3.4. PIA.....37
3.4.1. DIAGNOSTICS PIA.....37
3.4.2. DIRECTION PIA.....38
3.4.3. KEYPAD PIA.....39
3.4.4. LED PIA.....40
3.4.5. ROTARY SWITCH PIA.....40
3.5. PTM.....41
3.6. SPEECH INTERFACE.....42
3.6.1. AUDIO INPUT.....42
3.6.2. SPEECH SYNTHESIS.....45

4. SOFTWARE.....48
4.1. MONITOR CONCEPT.....48
4.1.1. KEYBOARD MONITOR.....48
4.1.2. MANUAL MONITOR.....50
4.1.3. DIAGNOSTICS MONITOR.....51
4.2. AUDIO INTERFACE.....53
4.2.1. ROOM NOISE.....53
4.2.2. AUDIO INPUT.....58
4.2.3. SPEECH SYNTHESIS.....59
4.3. MANUAL MOVEMENT.....59

5. RESULTS.....63
5.1. DISTANCE ERRORS.....63
5.2. DOWNWARD FORCE.....65

6. FUTURE DEVELOPMENTS.....66
6.1. SPEECH RECOGNITION.....66
6.1.1. DYNAMIC TIME WARPING.....66
6.1.2. TIME NORMALIZATION.....72
6.2. CLOSED LOOP CONTROL.....75
6.2.1. PURPOSE.....75
6.2.2. CURRENT SENSING.....76

7. CONCLUSION.....78

8. REFERENCES.....79

Appendices

A. SERVICE MANUAL.....83
A.1. DIAGNOSTIC TEST CONCEPT.....83
A.2. DIAGNOSTIC DEBUG TESTS.....83
A.2.1. DIAGNOSTIC DISPLAY TEST.....83
A.2.2. LED TESTS 1 - 2.....84
A.2.3. PERIPHERAL INTERFACE ADAPTERS.....85
A.2.3.1. PIA TESTS 1 - 3.....85
A.2.3.2. PIA TESTS 4 - 6.....85
A.2.3.3. PIA TESTS 7 - 9.....86
A.2.4. PROGRAMMABLE TIMER MODULES.....87
A.2.4.1. PTM TESTS 1 - 3.....87
A.2.4.2. PTM RESET TEST.....87
A.2.5. RANDOM ACCESS MEMORY.....88
A.2.5.1. RAM TESTS 1 - 4.....88
A.2.5.2. RAM TESTS 5 - 8.....88
A.2.6. SPEECH-SYNTHESIS TEST.....89
A.2.7. STEPPER MOTORS.....89
A.2.7.1. STEPPER MOTOR DIRECTION TESTS.....89
A.2.7.2. STEPPER MOTOR MOVEMENT TESTS.....89
A.2.7.3. STEPPER MOTOR PULSE DETECTION TESTS.....90
A.2.8. SWITCHES.....91
A.2.8.1. DISH ROTATION SWITCH TEST.....91
A.2.8.2. LEARN OPERATE SWITCH TEST.....92
A.2.8.3. MANUAL OPERATION SWITCH TEST.....93
A.2.8.4. STANDBY SWITCH TEST.....93
A.2.8.5. WAIT SWITCH TEST.....94
A.3. DIAGNOSTIC PROBE TESTS.....95
A.3.1. OSCILLATORS.....95
A.3.1.1. BAUD RATE TEST.....95
A.3.1.2. BAUD RATE GENERATOR CRYSTAL TEST.....96
A.3.1.3. MICROPROCESSOR CRYSTAL TEST.....97
A.3.1.4. REAL TIME CLOCK CRYSTAL TEST.....97
A.3.2. SWITCHES.....97
A.3.2.1. DISH ROTATION SWITCH TEST.....97
A.3.2.2. LEARN/OPERATE SWITCH TEST.....98
A.3.2.3. STANDBY SWITCH TEST.....99
A.3.2.4. WAIT SWITCH TEST.....100
A.4. DIAGNOSTIC DISPLAY VALUES.....101

B. OPERATOR'S MANUAL.....102
B.1. AUTOMATIC FEEDING OPERATION.....102
B.2. MANUAL OPERATION.....103
B.2.1. KEYPAD OPERATION.....103
B.2.2. AUDIO OPERATION.....104
B.3. ADDITIONAL FEATURES.....105
B.4. SWITCH SETTING SUMMARY.....105
B.5. KEYPAD SUMMARY.....106

C. CIRCUIT DIAGRAMS.....107

LIST OF TABLES

1-1: COST SUMMARY.....2

5-1: WEIGHT (gms) VS DISTANCE ERROR (mms) 63
5-2: WEIGHT (gms) VS DISTANCE ERROR (%) 64
5-3: MAXIMUM DOWNWARD FORCE (lbs).....65

A-1: LED-PIA CONNECTIONS.....84
A-2: PIA-EC CONNECTIONS.....86
A-3: PTM TIMER OUTPUTS.....87
A-4: RAM TEST ICs.....88
A-5: STEPPER MOTOR-EC CONNECTIONS.....90
A-6: DISH ROTATION SWITCH-PIA CONNECTIONS.....91
A-7: WAIT SWITCH-EC CONNECTIONS.....95
A-8: DISH ROTATION SWITCH POLES-PIA CONNECTIONS.....98
A-9: LEARN/OPERATE SWITCH-EC CONNECTIONS.....99
A-10: STANDBY SWITCH-EC CONNECTIONS.....100
A-11: WAIT SWITCH POLE-PIA CONNECTIONS.....100

LIST OF FIGURES

1-1: BLOCK DIAGRAM OF ROBOTIC MANIPULATOR SYSTEM.....3

2-1: STEPPER MOTOR LOGIC.....7
2-2: ZERO CROSSING WAVEFORMS.....9
2-3: ZERO CROSSING DETECTOR.....10
2-4: LEVEL SHIFTER WAVEFORMS.....11
2-5: LEVEL SHIFTER.....12
2-6: PHASE SEQUENCE GENERATOR WAVEFORMS.....14
2-7: REDUCED PHASE WAVEFORMS.....15
2-8: PHASE SEQUENCE GENERATOR STATE GRAPH.....16
2-9: PHASE SEQUENCE GENERATOR STATE TABLE.....16
2-10: PHASE SEQUENCE GENERATOR KARNAUGH MAPS.....17
2-11: STATE GRAPH FOR DIRECTION DETECTION.....18
2-12: NEXT STATE TABLE FOR DIRECTION DETECTION.....19
2-13: REDUCED STATE TABLE FOR DIRECTION DETECTION.....19
2-14: NEXT STATE TABLE FOR DIRECTION DETECTION.....20
2-15: S-R FLIP-FLOP CHARACTERISTICS.....20
2-16: S-R FLIP-FLOP A.....21
2-17: S-R FLIP-FLOP B.....21
2-18: SET-RESET FOR FLIP-FLOP A.....21
2-19: SET-RESET FOR FLIP-FLOP B.....22
2-20: OUTPUT FOR DIRECTION DETECTION.....22
2-21: COORDINATE AND LINK ORIENTATION.....26
2-22: TWO LINK MANIPULATOR IN X-Y SPACE.....29
2-23: MANIPULATOR DIFFERENCE.....31

3-1: SINGLE PRECISION FORMAT.....35
3-2: BLOCK DIAGRAM FOR AUDIO INPUT.....43
3-3: A/D GAIN.....44

4-1: KEYBOARD MONITOR FLOWCHART.....49
4-2: MANUAL MONITOR FLOWCHART.....51
4-3: DIAGNOSTICS MONITOR FLOWCHART.....52
4-4: SIGNAL FOR NOISE LEVEL DETERMINATION.....54
4-5: NOISE ROUTINE.....55
4-6: MOVE RIGHT FLOWCHART.....60

6-1: EXAMPLE TEMPLATES.....67
6-2: DISTANCE MEASURE MATRIX.....69
6-3: LOCAL DECISION.....70
6-4: DYNAMIC PROGRAMMING MATRIX.....71
6-5: EXAMPLE UTTERANCE.....75
6-6: DIFFERENCE UTTERANCE.....75
6-7: COMPRESSED UTTERANCE.....75
6-8: CURRENT WAVEFORMS.....77

B-1: KEYPAD LAYOUT.....104

C-1: BOARD A, BOARD LAYOUT.....107
C-2: BOARD A, MICROPROCESSOR AND RESET.....108

C-3: BOARD A, BUFFERS.....109
C-4: BOARD A, ADDRESS DECODING.....110
C-5: BOARD A, USER RAM.....111
C-6: BOARD A, HOST ACIA.....112
C-7: BOARD A, TERMINAL ACIA.....113
C-8: BOARD A, SELECTABLE BAUD RATE GENERATOR.....114
C-9: BOARD A, PTM 3.....115
C-10: BOARD A, USER ROM.....116
C-11: BOARD B, BOARD LAYOUT.....117
C-12: BOARD B, SPEECH PROCESSOR AND AUDIO AMPLIFIER....118
C-13: BOARD B, REAL TIME CLOCK & FLOATING POINT ROM....119
C-14: BOARD B, AUDIO INPUT.....120
C-15: BOARD B, SAMPLE AND HOLD + ANTIALIASING FILTER...121
C-16: BOARD B, HIGH PASS FILTER FOR MICROPHONE.....122
C-17: BOARD B, PROGRAMMABLE AGC.....123
C-18: BOARD B, PIA 5.....124
C-19: BOARD B, PIA 6.....125
C-20: BOARD B, PIA 7.....126
C-21: BOARD B, PIA 8.....127
C-22: BOARD C, BOARD LAYOUT.....128
C-23: BOARD C, PTM 2.....129
C-24: BOARD C, PTM 1.....130
C-25: BOARD C, DIAGNOSTIC DISPLAY.....131
C-26: BOARD C, PIA 1.....132
C-27: BOARD C, PIA 2.....133
C-28: BOARD C, PIA 3.....134
C-29: BOARD C, PIA 4.....135
C-30: BOARD C, STEPPER MOTOR SELECT.....136
C-31: MEMORY MAP.....137
C-32: RIBBON CABLE CONNECTORS.....138
C-33: D CONNECTOR FOR STEPPER MOTORS.....139

LIST OF ABBREVIATIONS

ACIA	Asynchronous communications interface adapter
AGC	Automatic gain control
BCD	Binary coded decimal
CMOS	Complementary Metal-Oxide Semiconductor
DIP	Dual inline package
DTW	Dynamic time warping
E	Enable clock
EC	Edge connector
EPROM	Erasable programmable read only memory
FIFO	First in, first out
FPCB	Floating Point Control Block
HPF	High-pass filter
Hz	Hertz
IRQ	Interrupt request
KHz	Kilohertz
LED	Light emitting diode
LSB	Least significant bit
LPC	Linear predictive coding
LPF	Low-pass filter
MHz	Megahertz
MSB	Most significant bit
NMI	Nonmaskable interrupt
NMOS	N-channel Metal-Oxide Semiconductor
PIA	Peripheral interface adapter
PTM	Programmable timer module
PWM	Pulse Width Modulation
Q	Quadrature clock
RAM	Random access memory
RC	Ribbon cable
ROM	Read only memory
TTL	Transistor transistor logic
V	Volts

1. INTRODUCTION

Unfortunately, there exists in today's high technology society a large number of people who are unable to care for themselves. These handicapped persons, through accidents, wars, crippling diseases, etc., are not capable of performing the simplest everyday tasks many people take for granted. A robotic device would allow these people some independence, while reducing their moral, financial, and physical dependence upon others.

The robotic arm described in this thesis (outlined in figure 1-1) is a versatile device capable of servicing a large variety of handicapped users. Primarily, it will provide the users with the opportunity to feed themselves. A teaching session will instruct the machine to scoop food from a bowl and bring it to the operator's mouth. Although the instruction process configures the system, reperforming the instruction phase will rededicate the device.

The versatility of such a device may be further increased through the addition of a manual operating system. This allows the user, through the use of a set of predefined commands, the ability to perform additional tasks. Manual operation is performed through either a keypad or audio input, and other interfaces may be used.

This device is to be designed and constructed as inexpensively as possible. Cost effective design will give

more handicapped users the opportunity to use the device. Stepper motors are employed, as opposed to servo motors, due to the difference in cost.

DESCRIPTION	COST
STEPPER MOTORS	$750
POWER SUPPLY	$500
MACHINING	$500
ELECTRONICS	$500

Table 1-1: COST SUMMARY

Although the system uses open loop control, opto-isolators are used as limit switches in order to prevent the motors from operating outside of a predefined range. Costs may be further minimized by using the motors as transducers. During the teaching phase, the waveforms developed in the windings are converted into pulse trains and stored by the microprocessor. Once again, common off the shelf circuitry is employed.

One microprocessor provides adequate control of the arm and avoids the additional costs of added processors along with the associated logic such as bus arbitrators. Robotic manipulators employ a varying number of joints. For this application, the five joints used are capable of providing a sufficient number of degrees of freedom. This reduction of the number of stepper motors is very significant, since some of the motors used here cost

almost two hundred dollars.

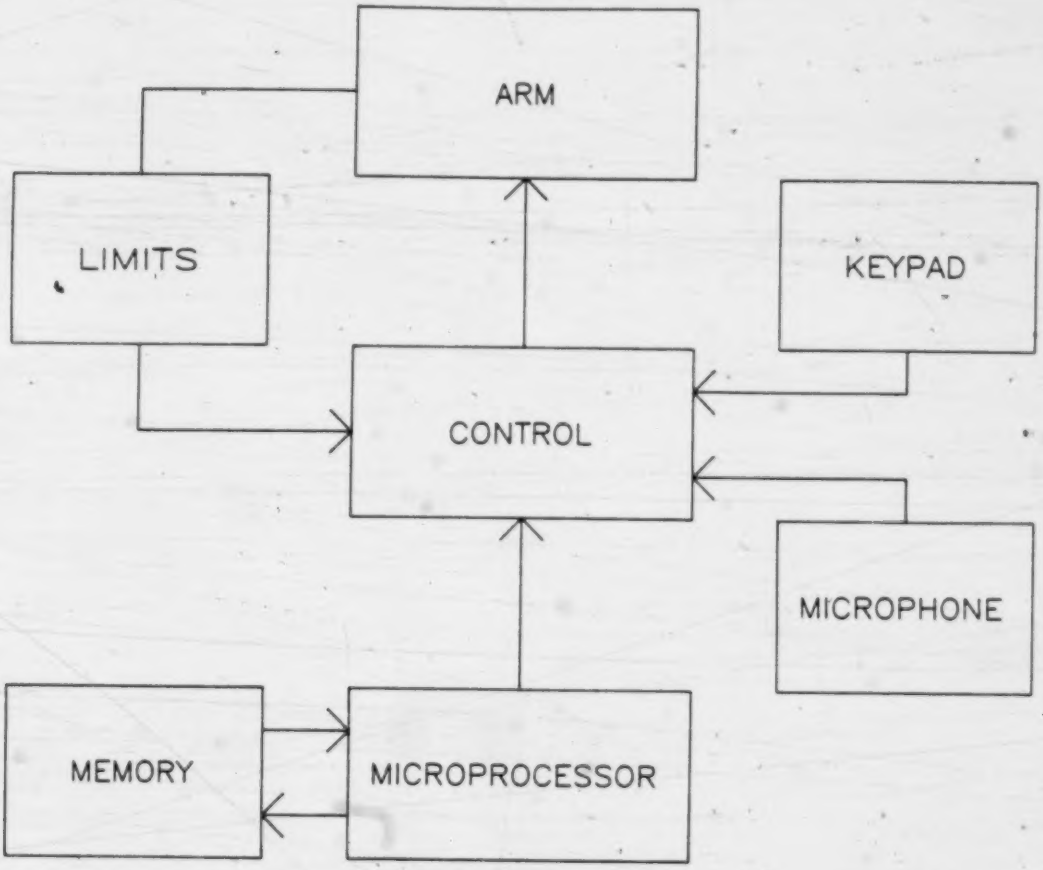


Fig. 1-1: BLOCK DIAGRAM OF ROBOTIC MANIPULATOR SYSTEM

In this thesis the design and implementation of a robotic manipulator for the handicapped is studied. Chapter two investigates the design problems encountered. These problems involve the logic and circuitry required to operate the stepper motors as transducers. In addition, the operation and speed regulation, of the stepper motors,

required to move the arm is investigated.

The second design question studied involves monitoring the manipulator as it travels through Cartesian space. The problem arises when one attempts to perform floating point operations efficiently. Efficiency must be achieved for both speed of execution and cost of implementation.

Chapters three and four investigate the hardware and software required to implement and operate the device. The software section details the use of the hardware and the implementation of the design concepts. The hardware chapter explains the functions of the circuitry and some of the rational used in selecting various components.

Chapter five summarizes two experiments which were performed in order to determine the limitations of the manipulator. The first experiment loaded the arm down with weights and tested its accuracy for various operating speeds. The second experiment investigated the manipulator's ability to generate downward force at various operating speeds.

The sixth chapter investigates two possible extensions to the system. Speech recognition maybe added as an additional remote interface. Two concepts, dynamic time warping and time normalization, are devloped for any future use.

The second possible extension is the addition of closed loop control to the stepper motors. One approach

translates mechanical movement into electrical signals, ie. optical detectors. Another method uses only the electrical signals developed by the stepper motors. This method is based on the interpretation of the current waveforms generated in the motor windings.

The first appendix is a service manual which maybe used to maintain the device. The service manual is divided into two types of tests, diagnostic and probe. Diagnostic tests are software routines which perform various tests on the system hardware. These tests use a set of eight switches and two seven segment displays, which are internally located. The probe tests require equipment to monitor and observe various points in the hardware. For example, the synchronization clocks from the microprocessor may be observed on a oscilloscope.

The second appendix is the operator's manual. This manual may also be subdivided into two sections. The first explains the procedure to utilize the automatic feeding mode. Both programming and playback of the device are explained. The second mode of operation involves manual control of the arm. This section instructs the user on how to use the remote devices, keypad and audio input.

2. DESIGN CONSIDERATIONS

2.1. STEPPER MOTORS

The five stepper motors, which constitute the robotic manipulator, may be used in two different operating modes (see figure 2-1). For the first mode, the stepper motors operate as transducers and provide angular displacement information to the system. During the alternate mode, phase information is used to drive the stepper motors.

Between the stepper motors and operating circuitry a relay controller is used to switch the windings to the appropriate circuitry. The relays permit the angular displacement information to pass to or from the corresponding logic, transducing or driving. In addition, relay control switches the stepper motors from 5 to 12 Volts. The lower voltage is used by the transducing circuitry as it simplifies the process of converting the phase signals to transistor-transistor logic (TTL) compatible levels. The driving circuitry uses the TTL phase information and the 12 Volt supply to step the motors. By using the 12 Volt supply, the maximum amount of torque is achieved at the motor shaft.

The first two stages (see figure 2-1) convert the phase information to TTL compatible signals. The zero crossing detector transforms the stepper motor signals to two discrete levels, +/- 5 Volts. The following stage, a

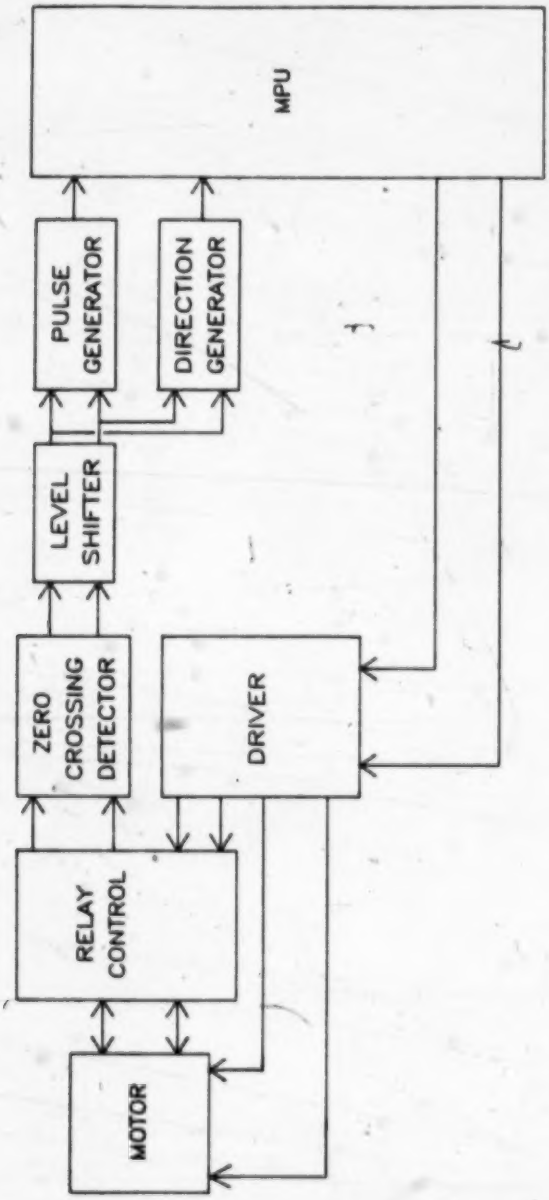


Fig. 2-1: STEPPER MOTOR LOGIC

level shifter, converts this signal so that it is TTL compatible.

The TTL compatible signals are split and used to drive two different detection devices. The direction detector uses the present and previous phase information to determine the direction of rotation. A single output line is used to indicate the direction of the last step, where a logic "1" indicates clockwise while a logic "0" indicates counterclockwise. The step detector produces a pulse when the stepper motor moves. The step detector uses a single output and produces one pulse per step.

The stepper motor driving circuitry features a phase sequence generator. The phase sequence generator uses two inputs, direction and clock pulse, to determine the next phase settings required by the stepper motor. The direction signal posseses the same characteristics as its transducer counterpart and sets the rotation orientation for the next step. The clock pulse is used to make the step occur at a specified moment.

2.1.1. ZERO CROSSING DETECTOR

Each of the robotic manipulator stepper motors requires two zero crossing detectors, one for each set of windings. Figure 2-2 illustrates the operation of this first stage in the detection of pulses and rotation direction. Figure 2-2a represents an incoming signal while figure 2-2b shows the output from the zero crossing

detector.

As illustrated, the zero crossing detector will drive the output to the appropriate rail, determined by the operational amplifier (OP-AMP), every time a sign change occurs in the incoming signal. For a positive to negative transition, the output attains the value of the negative portion of the power supply and remains constant until the next transition. The opposite applies for the negative to positive transition. Since the incoming signal is supplied from a stepper motor, each positive to negative transition will be followed by a negative to positive transition, and vice versa.

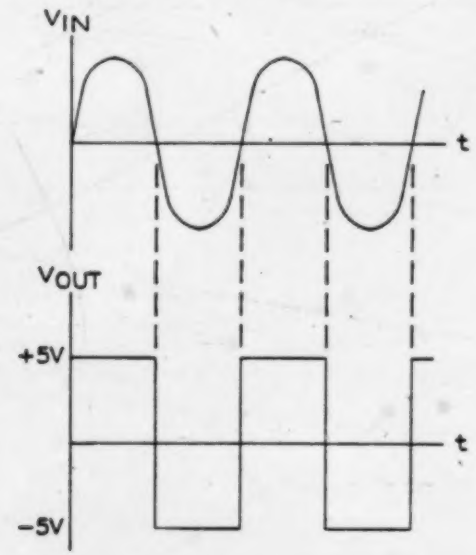


Fig. 2-2: ZERO CROSSING WAVEFORMS

The two diodes (see figure 2-3), D1 and D2, protect

the OP-AMP from an excessive input voltage. This is accomplished by clamping the input to either +/- Vd, the voltage drop across a diode while it is conducting. Inexpensive, readily available diodes are used, with typically Vd ≅ 0.7 Volts (V.), well within the allowable range.

In order to protect the diodes, the resistor, R1, is used. This component will limit the current through the diodes. The resistor, R2, prevents offset problems, thus insuring that the zero crossing detector references ground and not some finite nonzero value.

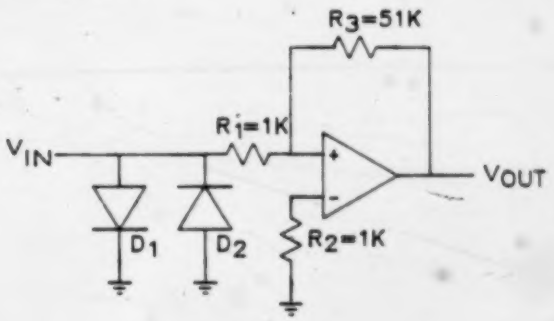


Fig. 2-3: ZERO CROSSING DETECTOR

The resistor, R3, provides feedback and serves two purposes. If VIN has a low frequency, the time required to cross 0 Volts will increase. Positive feedback insures that VOUT can switch quickly from one saturation voltage to the other.

If any noise exists at the input, the output, VOUT, may fluctuate between the two saturation levels. Thus zero reference crossings are detected for noise as well as VIN. Positive feedback prevents the detection of noise transition changes.

2.1.2. LEVEL SHIFTER

The level shifter converts the signal received from the zero crossing detector. Since the transducing and driving circuitry is constructed using digital logic, the signal must be manipulated so that a logic "0" corresponds

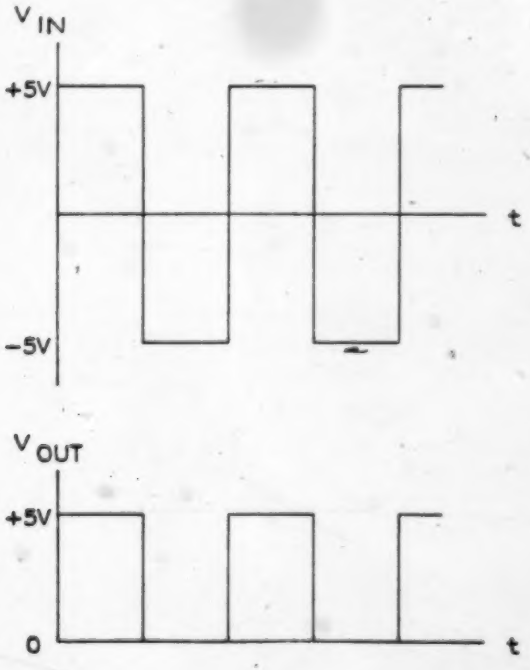


Fig. 2-4: LEVEL SHIFTER WAVEFORMS

to ground. The resulting signal is TTL compatible but may also be used to drive complementary metal-oxide semiconductor (CMOS) logic.

Figure 2-4 illustrates the input and output signal used by the level shifter. Note that the output mirrors the input for both frequency and zero crossing occurrences. The difference occurs in the peak to peak magnitude. The negative portion of the input has been clipped and set to ground.

The level shifter is designed using a type 741 OP-AMP and it is illustrated in figure 2-5. The inverting input is pulled directly to ground and therefore the lower threshold is :

$$V_{LT} = 0 \qquad (2\text{-}1)$$

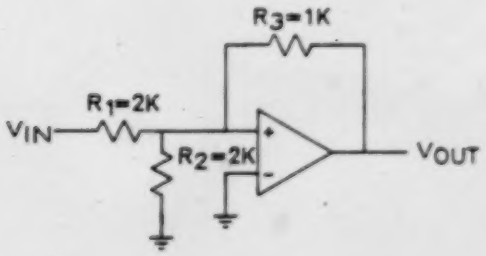


Fig. 2-5: LEVEL SHIFTER

The upper threshold is determined by the positive power supply and is:

$$V_{UT} = 5 \text{ Volts} \qquad (2\text{-}2)$$

At the output, the maximum and minimum is regulated by these saturation values.

Referring to figure 2-5, let the voltage at the noninverting input be VIO. At this point, VIO is determined from the voltage divider constructed from the resistor series, R1 and R2. Therefore, at the noninverting input :

$$VIO = \frac{R1}{R1 + R2} VIN = \frac{1}{2} VIN \qquad (2\text{-}3)$$

For the positive feedback amplifier, the overall gain can be reduced to :

$$A = \frac{R3}{RI} \qquad (2\text{-}4)$$

The resistance RI is equivalent to the resistors which constitute the voltage divider :

$$RI = \frac{R2}{R1 + R2} \qquad (2\text{-}5)$$

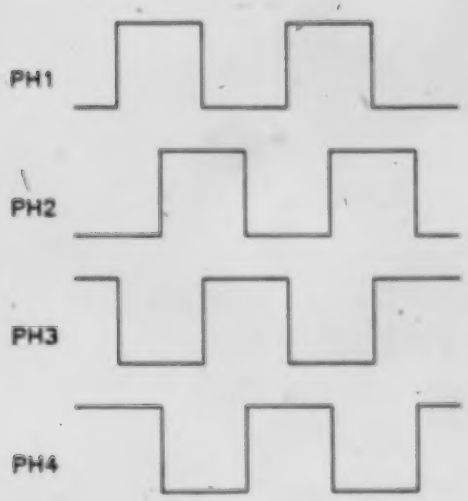
Substituting into the gain equation yields :

$$A = \frac{1}{1/2} = 2 \qquad (2\text{-}6)$$

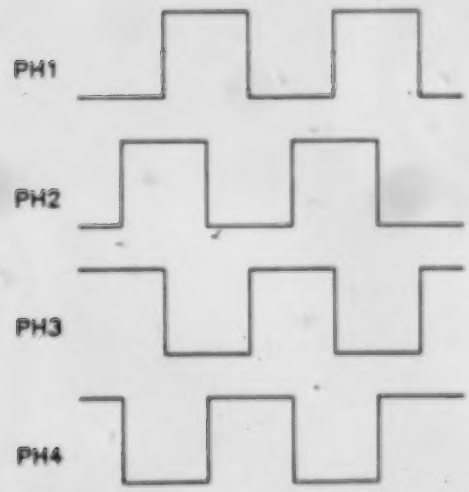
The level shifter has transformed a +/- 5 Volt signal to 0 - 5 Volts. During this process, the dominant signal features are retained. The stepper motor phase signals are now ready to be monitored using digital logic.

2.1.3. PHASE SEQUENCE GENERATOR

For the four phase stepper motors, two phase



f=1	
PH1	01100
PH2	00110
PH3	10011
PH4	11001



f=0	
PH1	01100
PH2	00110
PH3	10011
PH4	11001

Fig. 2-6: PHASE SEQUENCE GENERATOR WAVEFORMS

excitation is used. Two phase excitation is characterized by four unique steps. Compared to other methods, for example eight step excitation, this four step excitation yields decreased angular resolution. Although four step excitation uses larger angular increments, the torque produced is increased. Letting f denote the direction of rotation, observe the excitation waveforms shown in figure 2-6.

For either direction, note that two phases are always excited while the other two are not; hence the term two phase excitation. Observe that phase 3 is always the complement of phase 1 and phase 4 is always the complement of phase 2. The phase sequence generator design is simplified by only considering the first two phases and complementing them for implementation.

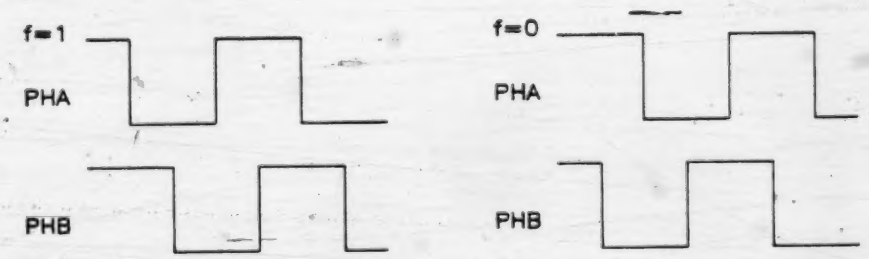


Fig. 2-7: REDUCED PHASE WAVEFORMS

The waveforms for the reduced phase criterion are

shown in figure 2-7.

Using the phase information, a state graph is constructed. From the state graph, figure 2-8, observe the two criteria required for a phase change (step of the motors), direction and a pulse. The pulse signal is a synchronization signal used to clock flip-flops.

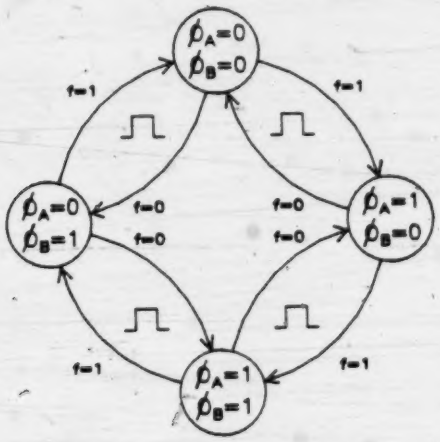


Fig. 2-8: PHASE SEQUENCE GENERATOR STATE GRAPH

$\phi_A \phi_B$	ϕ_A^+ F=0	ϕ_B^+ F=0	ϕ_A^+ F=1	ϕ_B^+ F=1
0 0	0	1	1	0
0 1	1	1	0	0
1 0	0	0	1	1
1 1	1	0	0	1

Fig. 2-9: PHASE SEQUENCE GENERATOR STATE TABLE

From the state graph, a "next-state" table is

constructed (see figure 2-9). Although only the direction signal is shown, the presence of the pulse signal will be implied for the duration of this derivation.

For the phase sequence generator, D flip-flops are used. To derive the flip-flop equations, first Karnaugh maps are drawn using the next state table (see figure 2-10).

For the first flip-flop :

$$DA = (\overline{f}\phi B + f\overline{\phi}B)\overline{\phi}A + (\overline{f}\phi B + f\overline{\phi}B)\phi A$$
$$= \phi B\overline{f} + \overline{\phi}Bf$$
$$= \phi B \oplus f \qquad (2\text{-}7)$$

For the second flip-flop :

$$DB = (\overline{f\phi}A + f\phi A)\phi B + (\overline{f\phi}A + f\phi A)\phi B$$
$$= \overline{\phi}A\overline{f} + \phi A$$
$$= \overline{\phi A \oplus f} \qquad (2\text{-}8)$$

$\phi_A\phi_B$ \ f	0	1
0 0	0	1
0 1	1	0
1 0	0	1
1 1	1	0

ϕ_A^+

$\phi_A\phi_B$ \ f	0	1
0 0	1	0
0 1	1	0
1 0	0	1
1 1	0	1

ϕ_B^+

Fig. 2-10: PHASE SEQUENCE GENERATOR KARNAUGH MAPS

2.1.4. DIRECTION DETECTION

When the stepper motors are operated as transducers,

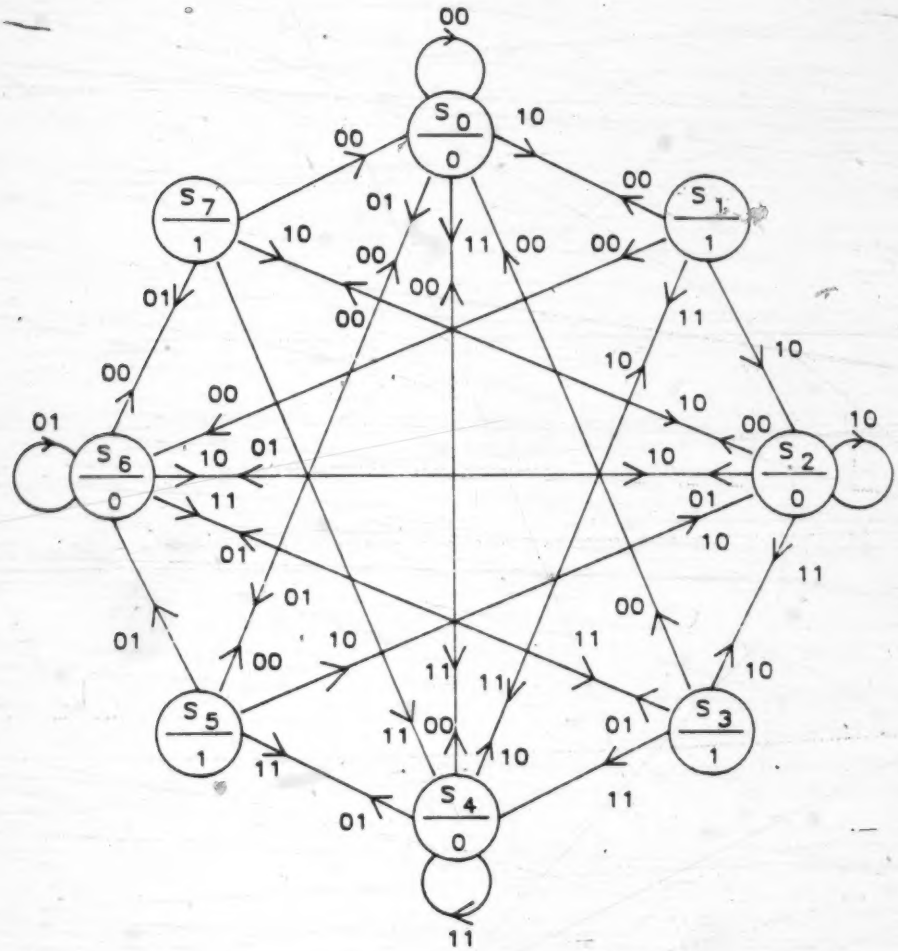


Fig. 2-11: STATE GRAPH FOR DIRECTION DETECTION

step and direction detection information can be extracted. Like the phase sequence generator, two windings are always excited while the remaining two are not. Changes in the pulses are used to indicate movement of the stepper motor. Direction detection is accomplished by monitoring the state of the windings and noting which windings lead and lag when a step occurs. The step and direction criterion

PRESENT STATE	NEXT STATE 0 0	0 1	1 1	1 0	PRESENT OUTPUT
S_0	S_0	S_5	S_4	S_1	0
S_1	S_0	S_6	S_4	S_2	1
S_2	S_7	S_6	S_3	S_2	0
S_3	S_0	S_6	S_4	S_2	1
S_4	S_0	S_5	S_4	S_1	0
S_5	S_0	S_6	S_4	S_2	1
S_6	S_7	S_6	S_3	S_2	0
S_7	S_0	S_6	S_4	S_2	1

Fig. 2-12: NEXT STATE TABLE FOR DIRECTION DETECTION

PRESENT STATE	NEXT STATE 0 0	0 1	1 1	1 0	PRESENT OUTPUT
S_0	S_0	S_6 S_1	S_4 S_0	S_1	0
S_1	S_0	S_6 S_2	S_4 S_0	S_2	1
S_2	S_7 S_1	S_6 S_2	S_3 S_0	S_2	0
~~S_6 S_7~~	S_0	~~S_6 S_7~~	~~S_4~~	~~S_2~~	1
~~S_6 S_0~~	S_0	~~S_6 S_1~~	~~S_4~~	~~S_1~~	0
~~S_6 S_1~~	S_0	~~S_6 S_2~~	~~S_4~~	~~S_2~~	1
~~S_6 S_2~~	S_7	~~S_6 S_2~~	~~S_3~~	~~S_2~~	0
~~S_7 S_1~~	S_0	~~S_6 S_7~~	~~S_4~~	~~S_2~~	1

Fig. 2-13: REDUCED STATE TABLE FOR DIRECTION DETECTION

is summarized with a "state graph" (see figure 2-11).

From the "state graph", an eight state "next state" table is constructed (see figure 2-12).

When the "next state" table (and "state graph") were developed, the inclusion of extra states was ignored. After the elimination of redundant and equivalent states, [7] a three state solution, for step and direction detection, is produced (see figure 2-13).

After state assignment, the next "state table", illustrated in figure 2-14, is derived.

PRESENT STATE AB	NEXT STATE A^+B^+ 0 0	0 1	1 1	1 0	PRESENT OUTPUT Z
00	00	01	00	01	0
01	00	11	00	11	1
11	01	11	01	11	0

Fig. 2-14: NEXT STATE TABLE FOR DIRECTION DETECTION

Q	Q^+	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Fig. 2-15: S-R FLIP-FLOP CHARACTERISTICS

Recall the characteristic behavior, for S-R flip-flops, illustrated on the following page.

PRESENT STATE AB	NEXT STATE 0 0 $S_A R_A$	0 1 $S_A R_A$	1 1 $S_A R_A$	1 0 $S_A R_A$	PRESENT OUTPUT Z
00	0 X	0 X	0 X	0 X	0
01	0 X	1 0	0 X	1 0	1
11	0 1	X 0	0 1	X 0	0

Fig. 2-16: S-R FLIP-FLOP A

PRESENT STATE AB	NEXT STATE 0 0 $S_B R_B$	0 1 $S_B R_B$	1 1 $S_B R_B$	1 0 $S_B R_B$	PRESENT OUTPUT Z
00	0 X	1 0	0 X	1 0	0
01	0 1	X 0	0 1	X 0	1
11	X 0	X 0	X 0	X 0	0

Fig. 2-17: S-R FLIP-FLOP B

AB	0 0	0 1	1 1	1 0
	$\phi_1\phi_2$			
0 0	0	0	0	0
0 1	0	1	0	1
1 1	0	X	0	X
1 0				

$S_A = B(\phi_1/\phi_2)$

AB	0 0	0 1	1 1	1 0
	$\phi_1\phi_2$			
0 0	X	X	X	X
0 1	X	0	X	0
1 1	1	0	1	0
1 0				

$R_A = \overline{\phi_1 \cdot \phi_2}$

Fig. 2-18: SET-RESET FOR FLIP-FLOP A

The "next state" table may be realized using two S-R flip-flops. The first flip-flop is realized in figure 2-16 and the second flip-flop is realized in figure 2-17.

In order to derive the flip-flop equations, the next "state tables" are transformed to Karnaugh maps (see figures 2-18 to 2-19).

AB \ $\phi_1\phi_2$	00	01	11	10
00	0	1	0	1
01	0	X	0	X
11	X	X	X	X
10				

$S_B = \phi_1 \bullet \phi_2$

AB \ $\phi_1\phi_2$	00	01	11	10
00	X	0	X	0
01	1	0	1	0
11	0	0	0	0
10				

$R_B = \overline{A(\phi_1 \bullet \phi_2)}$

Fig. 2-19: SET-RESET FOR FLIP-FLOP B

Finally, the output is realized with the Karnaugh map shown in figure 2-20.

B \ A	0	1
0	0	
1	1	0

$Z = A\bar{B}$

Fig. 2-20: OUTPUT FOR DIRECTION DETECTION

2.1.5. SPEED CONTROL

Both modes of operation, "manual" and "programmed", can be used with full robotic manipulator speed control. Speed control is provided from a four pole rotary switch located in the upper left corner on the control panel. Although speed is regulated through a common device, different methods provide the actual control for each mode of operation.

The four pole rotary switch is interfaced to the system through the four higher order interface lines on port B of peripheral interface adapter (PIA) 4. The four distinct speed settings provided allow the user a sufficient degree of control. For each mode of operation, there is an ideal speed range in which the manipulator must operate.

For "manual" operation, a large number of floating point operations are being performed. Due to the long execution times, only the upper portion of the speed control range is required. The lower speed settings cause the manipulator to move at an unacceptably slow rate.

Unlike "manual" operation, the "playback" mode can operate at a much increased rate, due to the absence of position checking. During "manual" operation, a large number of floating point calculations are required to monitor the manipulator tip. The finite amount of time required for this position checking occurs between stepper

motor movements. For "playback" operation, the rate of operation is dependent upon the rate at which the arm was programmed and no lengthy calculations are performed between steps.

When a "programmed" motion is performed, stepper control is derived from dedicated programmable timer module (PTM) timer outputs. The driving circuitry, for each stepper motor, is clocked by a separate timer. The five dedicated timers are operated in 16 bit "continuous" mode with an external clock.

It is this external clock which provides the actual speed regulation. The remaining timer provides the external clock signal. The speed control switch determines the initialization value for this timer. Therefore, by varying the external clock timer, speed regulation is achieved.

Unlike "programmed" operation, "manual" operation is performed independent of PTM control. Instead, the stepper motor driving circuitry is controlled by a software clock. A variable double loop delay routine is executed after each step, thus preventing any other steps occurring. The speed control switch determines the length of the delay.

2.2. COORDINATE CALCULATIONS

For accurate movement along an axis, it is necessary to monitor the manipulator tip in Cartesian space. This is accomplished by using two sets of criteria, the known

structure of the manipulator and the monitored angular joint displacements. [9] Figure 2-21 illustrates the coordinate orientation, labeling of link lengths and labeling of joint angles.

$$X = L2COS(A1)COS(A2) + L4COS(A4) + L5SIN(A5)SIN(A4) - L6SIN(A2) \quad (2\text{-}9)$$

$$Y = L2COS(A1)SIN(A2) + L4SIN(A4) - L5SIN(A5)COS(A4) + L6COS(A2)$$

$$Z = L1 + L3 + L2SIN(A1) - L5COS(A5)$$

Define the following set of variables to simplify the coordinate equations.

$$A4 = A2 + A3 \quad (2\text{-}10)$$

$$L2 = L2COS(A1)$$

$$L5 = L5SIN(A5)$$

$$L13 = L1 + L3$$

After substitution, the following set of equations is obtained.

$$X = L2COS(A2) + L4COS(A4) + L5SIN(A4) - L6SIN(A2) \quad (2\text{-}11)$$

$$Y = L2SIN(A2) + L4SIN(A4) - L5COS(A4) - L6COS(A2)$$

$$Z = L13 + L2SIN(A1) - L5COS(A5)$$

There are several different approaches to calculating the coordinate equations. A quick and efficient approach would be to use a math coprocessor, such as Motorla's 68881. However this would not be

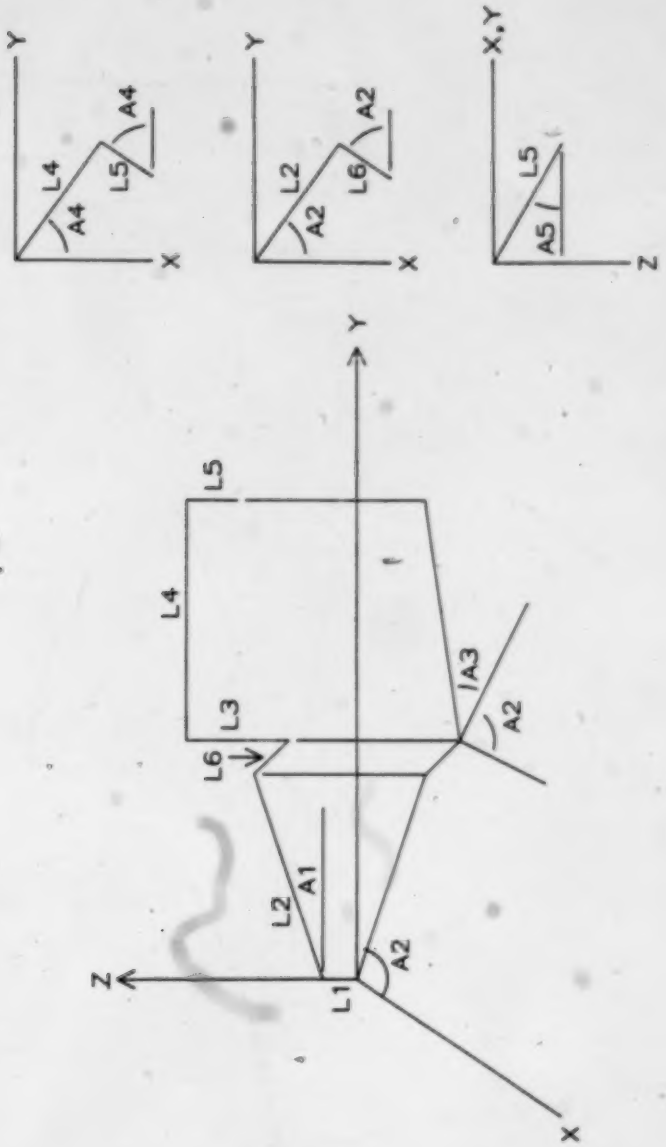


Fig. 2-21: COORDINATE AND LINK ORIENTATION

feasible, since it is priced in excess of three hundred dollars. Also, the 68881 is designed to operate with the 68000 series and this would mean overkill for this system.

Instead of Motorola's math coprocessor, National Semiconductor's math processor could be used. This device is capable of operating as a stand alone device or as a microprocessor peripheral. This device operates at only 400 KHz and requires the addition of a -4 Volt supply. In addition, there would an increase in hardware expenditure, for example first in first out (FIFO's), due to interfacing problems.

The third solution is to use packaged floating point subroutines. These are readily available in Motorola's 6839 floating point read only memory (ROM). No specialized hardware is required, since this device was designed to interface directly to the 6809's bus.

The 6839 floating point ROM was selected for several reasons. The ROM is inexpensive and requires no additional hardware expenditures. Although this method is incapable of high speed performance, it is not crucial for this application.

Although the 6839 can do all of the basic floating point math operations, the sine and cosine functions must be accounted for. The sin function can be represented as a series. For a four term series, six multiplication operations are required, requiring 2800 to 3200

microseconds each. Although this method did work, the execution time required caused the manipulator to move at an unacceptablely slow rate.

$$SIN(x) = \frac{x}{1} - \frac{x3}{3!} + \frac{x5}{5!} - \frac{x7}{7!} \qquad (2\text{-}12)$$

A second solution was to replace the series with look up tables, sacrificing memory for speed. Now each function call is executed by merely locating a value in a table. This method improved manipulator speed by several orders of magnitude.

For increased performance, only the position of the wrist joint not the manipulator tip is monitored. This procedure removes the link L5 from the equations.

$$X = L2COS(A2) + L4COS(A4) - L6SIN(A2) \qquad (2\text{-}13)$$

$$Y = L2SIN(A2) + L4SIN(A4) + L6COS(A2)$$

$$Z = L13 + L2SIN(A1)$$

The sine and cosine functions are performed using look up tables stored in ROM. For the stepper motors involved, either 0.25 O/step or 0.30 O/step increments are used. This implies the use of two look up tables. Joint angle A2 is a multiple of 0.25O while A3 is a multiple of 0.25O. The functions SIN(A2 + A3) and COS(A2 + A3) can not be easily calculated. The following trigonometric identities convert A4 calculations into separate A2 and A3 functions.

COS(A2 + A3) = COS(A2)COS(A3) - SIN(A2)SIN(A3) (2-14)

SIN(A2 + A3) = SIN(A2)COS(A3) + COS(A2)SIN(A3)

After substituting the identities, the following equations are arrived at.

X = L2COS(A2) + L4COS(A2)COS(A3) (2-15)

- L4SIN(A2)SIN(A3) - L6SIN(A2)

Y = L2SIN(A2) + L4SIN(A2)COS(A3)

+ L4COS(A2)SIN(A3) - L6SIN(A2)

Z = L13 + L2SIN(A1)

Now consider a two link manipulator operating in the x-y plane (see figure 2-22). It is desired to move from y1 to y2 yet maintain the x coordinate. Since the stepper motors can only rotate at a set increment, it is not always possible to achieve x = x1. Therefore, the

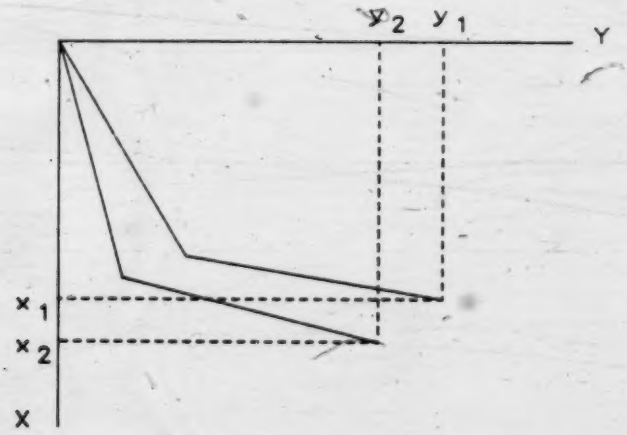


Fig. 2-22: TWO LINK MANIPULATOR IN X-Y SPACE

manipulator must be moved as close to x1 as possible.

First, the direction of movement is determined by subtracting x2 from x1. The sign of the difference implies the required direction of movement. The appropriate movement is performed and the difference between x1 and x2 is recalculated. This new difference falls into one of three categories:

a) the sign is the same as before.

b) the difference is zero.

c) the sign has changed.

Should the difference be zero, the tip is at the desired location and no further correction is required. However, this seldom occurs and the other two cases must be dealt with. If the sign is the same as before, the tip has not yet reached the desired position. The tip is again moved and the difference retested. This process is repeated until the difference is zero or a sign change occurs.

When a sign change occurs, the desired position lies between the current and previous point (see figure 2-23). The two differences, dx3 and dx2 are compared to determine which is the smallest. By moving to the point corresponding to the smallest difference, the manipulator tip position corrected as accurately as possible.

This method was implemented and found to perform unacceptably slow. Instead, a more efficient variation of

this algorithm was used. The movement until a sign change occurs is necessary and still performed. However, it is at this point that the two algorithms differ. Instead of minimizing the difference, only a sign change is searched for.

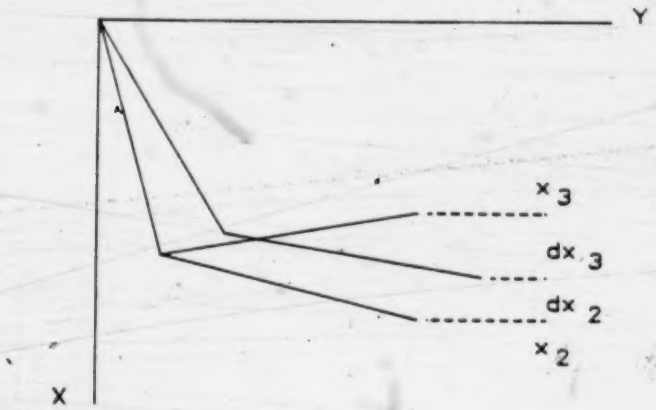


Fig. 2-23: MANIPULATOR DIFFERENCE

Once the tip has reached x3, the x correcting process is finished. To insure that the tip does not stray, with additional movements, the original value, xl, is restored as the current position. After the next step, the tip will try to reach xl again. Until a command for a different direction is received, the tip will always try to maintain xl.

This algorithm was implemented and found to operate

faster. The improved procedure requires significantly fewer floating point calculations. The decreased calculating produces the speed increase since one addition operation requires a minimum of 2400 microseconds. Addition executes the fastest out of the four major arithmetic operations.

Recall that the linkage L5 was ignored from coordinate calculations. This causes the wrist joint, not the manipulator tip, to be monitored. Motion in the X-Y plane, does not produce any deviations along the z axis. For z direction movement (up and down), the position of the wrist will vary an insignificant amount. The z coordinate will not be monitored and the coordinate calculations are reduced to:

$$X = L2COS(A2) + L4COS(A2)COS(A3) - L4SIN(A2)SIN(A3) - L6SIN(A2) \quad (2\text{-}16)$$

$$Y = L2SIN(A2) + L4SIN(A2)COS(A3) - L4COS(A2)SIN(A3) - L6SIN(A2)$$

3. HARDWARE

3.1. ACIA

The ASSIST09 monitor requires two asynchronous communications interface adapters (ACIA), referred to as the "host" and "terminal" (see figures C-6 and C-7). The "terminal" ACIA is connected to a "dumb" terminal and provides two functions. The monitor commands and services are accessed through the terminal. The "host" ACIA is used to communicate to a "host" computer. The "load" and "punch" commands permit downloading of assembled code directly into the system random access memory (RAM) from the "host". The source code is assembled on an IBM PC/XT using the Avocet 6809 cross assembler.

Both ACIAs communicate using the RS232C interfacing standard. Since no handshaking is performed with external devices, the two control lines, data carrier detect and clear to send, are ignored and tied to ground. For both ports, only the transmit and receive lines are used.

Two additional features, associated with transmit and receive, should be noted. First, all four lines may be visually monitored by light emitting diodes (LED) located on board A. At the DB-25 connector, transmit and receive have been switched, for the "terminal" ACIA only, allowing the use of a straight through cable. Since a modem was originally used, the "host" ACIA does not have the interchanged transmit and receive.

Both ACIAs are clocked by a common external source. The baud rate is produced by a Motorola MC14411 bit rate generator (see figure C-8). This device uses a frequency divider format to produce a range of 16, clock source based, output frequencies. The clock source is provided by a 1.8432 MHz crystal.

The previously mentioned output frequencies are based on one of four time bases; x1, x8, x16 and x64. The time base is selected by two rate select pins, RSA and RSB. For this application, RSA is pulled low and RSB is held high, the x16 time base is used. The appropriate output is selected by the two eight switch DIPs. The ASSIST09 monitor sets the ACIAs to divide the baud rate by 16. To achieve 9600 baud, 153 KHz must be selected from the bit rate generator.

3.2. FLOATING POINT ROM

The MC6839 floating point ROM is a Motorola standard product preprogrammed ROM. Since the device is unique to Motorola, this ROM does not conform to the bytewyde memory concept and comes in a nonstandard 24 pin, 8K dual inline packages (DIP). All of the supported operations, modes and precisions, conform to the IEEE proposed standard for binary floating arithmetic draft 8.0. Software implemented floating point operations are less expensive than hardware.

The routines are position independent and require a

contiguous 8K block of memory. Operation is not dependent upon internal registers or intermediate results. Three different precision modes are supported; single, double and extended.

All floating point operations are performed using normalized numbers in single precision format, represented by a 32 bit string (see figure 3-1). The layout for a single precision number, the format used to convert the encoded string to base ten, is illustrated below :

$$X = (-1)S * 2(\text{exponent} - 127) * (1.\text{significand}) \quad (3\text{-}1)$$

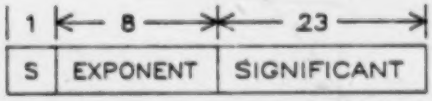


Fig. 3-1: SINGLE PRECISION FORMAT

The first bit, s, is the sign of the significand. The 8 bit exponent is biased by 127. Biasing the exponent allows easier floating point unsigned integer comparisons since condition codes can be calculated. The 23 bit significand yields a 24 bit digit in the range

$$1.0 < \text{significand} < 2.0 \quad (3\text{-}2)$$

Floating point operations are executed using either register or stack calls. Since intermediate results use the stack as a local storage area, an adequate stack size must be provided to prevent stack overflow. In comparison,

register calls require a maximum of 170 bytes while stack calls require 200 bytes.

The register calling sequence was chosen in order to save memory, 30 bytes of stack. This procedure uses the two index registers, user stack pointer and accumulator D. The addresses of the arguments are contained in the user stack pointer and Y index register, respectively. The address of the result is returned in the X index register while accumulator D contains the address of the floating point control block (FPCB). The register contents are retained upon completion. [6]

3.3. MEMORY

All of the memory (random access (RAM), read only (ROM) and erasable programmable read only (EPROM)) is designed using Mostek's bytewyde concept. The bytewyde concept features interchangeability due to a common pin configuration. This provides the flexibility to develop the software in RAM and replace it with an EPROM when it is operational. Device interchangeability leads to memory upgradeability for both 24 and 28 pin DIP packages. For example, to upgrade EPROM, replace a 4 kilobyte (2732A) with an 8 kilobyte (2764).

The bytewyde concept is adhered to by the majority of memory manufacturer's (with Motorola being the exception). The use of a standardized, upgradeable configuration prevents the memory system from becoming

obsolete. The bytewyde memory family is designed for compatibility with future devices. Finally, when compared to another standard, such as the unique one used by Motorola, the use of bytewyde memory products is significantly less expensive.

Two control lines are required for the EPROMs, chip enable and output enable. Since chip enable control is independent of bus contention, it is used for device selection. For a selected device, output enable control is used to gate data to the bus.

For the bidirectional devices, bytewyde RAMs, a third control line is required. Write enable control is used when writing to the RAMs and output enable control is used when reading. For edge activated devices, synchronization with the microprocessor clocks is required. Addresses are valid with the leading edge of the quadrature (Q) clock. For the enable (E) clock, data is valid on the rising edge and latched on the falling edge. The Q clock always leads the E clock by on half of one E clock cycle. The generation of the output enable and write enable control signals are shown in figure C-5. For proper RAM interfacing, the read and write signals must be synchronized with the appropriate clock.

3.4. PIA

3.4.1. DIAGNOSTICS PIA

Both ports of PIA three are dedicated to the ROM

resident diagnostic routines. Port A is configured as inputs and monitors an eight switch diagnostic DIP (see figure C-28). Since the switches are normally open, the interface lines must be set to a default logic level. This is accomplished through the 10K pull up resistors, setting the interface lines to a logic "1".

Port B is configured as outputs and is used to control two common anode seven segment displays. Each display is driven by a binary coded decimal (BCD) to seven segment decoder (see figure C-25). Between each cathode and open collector decoder output, a current limiting resistor is placed. Decimal numbers from zero to ninety nine can be shown on the displays. (For information on the use of the diagnostic display refer to the service manual located in appendix A.)

3.4.2. DIRECTION PIA

The stepper motor direction of rotation is monitored by PIA 1 located on board C (see figure C-26). Port A is configured as inputs and used to monitor the direction during the instruction phase. Port B is used to select the direction of rotation during playback and manual operation.

Separate input and output direction lines are required due to the circuitry generating these signals. Although only the first stepper motor is referred to in the following discussion, the argument applies to all five

manipulator motors. During the learning phase, the motion logic for stepper motor one uses the phase information to determine which direction the motor is rotating in. During playback or manual operation, the phase sequence generator must use the direction of rotation and current phase data to generate the next set of phase settings.

3.4.3. KEYPAD PIA

Nonvocal manual operation is provided through a four by four, sixteen button keypad. (Refer to the operator's manual for information on the button functions.) The keypad is based upon the matrix code configuration. A matrix code system is characterized by the absence of a common output pin. Instead, a sixteen entry grid is developed using four rows and four columns.

The four rows and four columns are ideally suited for interfacing to a PIA port. Port B of PIA 6 is used to interface the keypad to the system. Only one modification is required to monitor the keypad and determine if and which button is pressed.

The software, keypad monitor, sets a row high and then tests the columns. If a button is pressed, the appropriate column will also be high. However, if no button in the current row is pressed, all columns must be at a logic "0". To insure that this criterion is met, the columns are pulled low through resistors. The resistors prevent the power supply from being short circuited to

ground.

3.4.4. LED PIA

There are seven LEDs which indicate various switch settings and modes of operation. "Standby", "learn" and "operate" are connected to Port A of PIA 2. The least significant interface lines for Port B, PIA 4 are used to control the "keypad", "automatic", "voice" and "manual" LEDs.

All of the LEDs are controlled using the same approach. The LEDs are connected in a common anode configuration. The anodes are tied to 5 Volts and the LEDs are illuminated by pulling the cathodes low. Due to the small current rating of the LEDs, current limiting resistors are placed in series with the cathodes.

3.4.5. ROTARY SWITCH PIA

Three rotary switches; "dish rotation", "wait" and "speed", are used. All of the switches are configured as four pole single throw rotaries. One and one half PIA ports are used to interface the four pole rotary switches.

All three switches are interfaced using ports from PIA 4. Port A is configured for the "dish rotation" and "wait" switches (see figure C-29). The upper four lines of Port B monitor the "speed" switch. The common line for all of the switches is grounded and all of the poles are pulled high. The line corresponding to the switch position will be at a logic "0" while the remaining three lines

will be at a logic "1".

3.5. PTM

Nine programmable timers are provided by three PTMs. One PTM is required by the ASSIST09 monitor for the trace function. The two remaining PTMs are required for the robotic manipulator control.

The two PTMs located on board C are used to clock the manipulator motors during playback and programming. Each motor has a PTM timer which measures the time between steps during the instruction process. For playback, the timers reproduce the duration between steps. The sixth timer is used as a reference and clocks the five motor timers.

Each timer has a high impedance TTL compatible clock input. The output from the reference timer is connected to the clock inputs of the motor timers (see figures C-23 and C-24). The five remaining timer outputs are not used and therefore left unconnected.

Each timer has an active low, high impedance, TTL compatible asynchronous gate input. A rising edge on a gate input is used to trigger the clocks. The manipulator gate inputs are connected to the motor pulse generator which produces a pulse for each step. To insure that a suitable rising edge is received, the gate inputs are grounded through pull down resistors.

Since these PTMs are used only during two modes of

operation, the reset is connected to a PIA interface line. This permits the timers to be reset from software when necessary, for example, prior to programming or playback. All timer interrupts are monitored by the fast interrupt request.

The third PTM, located on board A, is configured for the ASSIST09 monitor. Like the previous PTMs, the gate inputs are pulled low through resistors (see figure C-9). The timers reference the system clock and therefore the gate inputs are not used.

Only one timer is used and is required by the monitor's "trace" command. The "trace" command uses nonmaskable interrupts generated by timer one. Since nonmaskable interrupts are active low, the output of timer one is complemented through an inverter. Since this PTM operates independent of manipulator control, both the reset and interrupt request are tied to the system.

3.6. SPEECH INTERFACE

3.6.1. AUDIO INPUT

The audio input hardware is based on General Instrument's SP1000. The SP1000 is an NMOS device which operates on a single 5 V supply with an on-board oscillator. The audio chip is TTL compatible, has a user programmable sampling rate and a standard microprocessor interface.

The SP1000 requires four additional building blocks

to produce a fully operational speech recognition system (see figure 3-02). First, a high-pass filter (HPF) for the microphone filters out all sounds below 250 Hz. The automatic gain controller (AGC) insures that the audio signal remains within the dynamic range of the A/D converter. The antialiasing filter removes high frequency transients, ie. aliases. Finally, an A/D converter is used to transmit the data to the SP1000.

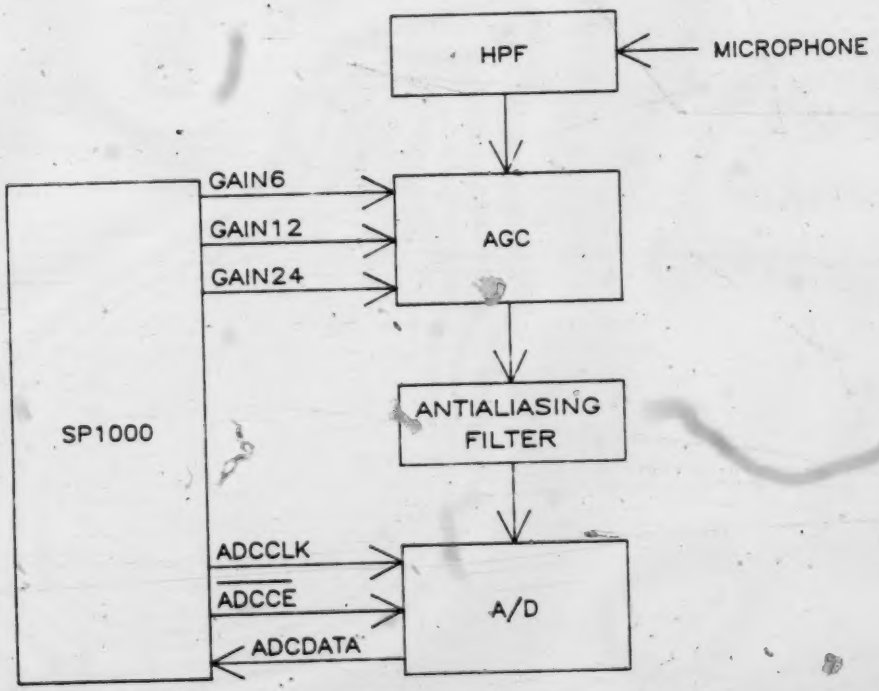


Fig. 3-2: BLOCK DIAGRAM FOR AUDIO INPUT

The software controlled sampling frequency ranges from 5.0 KHz to 15.9 KHz for audio input. An on-board automatic gain controller (AGC) allows the SP1000 to control an external input amplifier. The three gain outputs, from the SP1000, are used to construct an automatic gain control amplifier (see figure C-17). The AGC insures that the audio signal is always within the dynamic range of the A/D converter. This is accomplished by changing the gain for a pair of series connected noninverting amplifiers. Each gain output controls an analog CMOS switch which is situated in parallel with a resistor.

GAIN PINS			GAIN
24	12	6	REQUIRED
0	0	0	0
0	0	1	6
0	1	0	12
0	1	1	18
1	0	0	24
1	0	1	30
1	1	0	36
1	1	1	42

Fig. 3-3: A/D GAIN

The output pins are referred to as "GAIN 6", "GAIN 12" and "GAIN 24" corresponding to signal levels of 6, 12

and 24 db.. By combining the gain outputs, a range of 0 to 42 db, with 6 db increments, can be achieved (see figure 3-03). Through the gain pins; the SP1000 can regulate the incoming audio signal.

Between the AGC and A/D converter resides an antialiasing filter (see figure C-15). The antialiasing filter removes high frequency transients produced by the analog switches in the AGC. The aliases are removed using a second order low-pass filter with a cutoff frequency of 3.2 KHz.

The SP1000 provides three interface lines (clock, chip enable and data in) for the A/D converter. The data in receives serial data from the A/D converter. The active low chip enable selects the A/D converter and controls an analog switch which serves as a sample and hold circuit. The active high clock is only valid when the chip enable is low. This 150 KHz gated clock is used to synchronize the data.

3.6.2. SPEECH SYNTHESIS

Speech synthesis is accomplished using the General Instrument, SPO256A-AL2, allophone based speech processor. Like the Votrax SC-01, the SPO256A-AL2 is a second generation speech synthesizer. Both speech processors are commonly used devices however General Instrument's is less expensive, by at least a factor of 5, and readily available, a commonly stocked item as opposed to a

specialty item.

The speech processor can perform three types of synthesis; LPC, formant and allophone (see Veltri for a complete description on these methods). The SPO256A-AL2 has been preprogrammed with 64 allophones. The output is characterized by a signal to noise ratio of 35 dB, a dynamic range of 42 dB and a frequency response with an upper bound of 5 KHz. The output is pulse width modulated (PWM) by a 40 KHz carrier. The carrier is removed by a 5 KHz low pass filter (see figure C-12). The output is amplified by a low power audio amplifier with a gain of 50.

Words are produced by sending allophone strings to the speech processor. The allophones are transmitted through a PIA port to the address lines on the speech device. Three control lines are used to handshake the speech processor.

The strobe enable line determines the mode of operation. If the strobe enable line is low, data is latched on a low to high transition on any address line. Instead, the strobe enable line is tied high and data latched when the address latch data line is strobed low. When this operating mode is used, the load request line is used to indicate the condition of the address input buffer.

Whenever the address latch data strobes low and

latches data, load request will go high indicating that the input data buffer is full. Allophones can not be sent to the speech processor during this time. When the load request line goes low, the input data buffer is empty and allophone transmission may resume. The input data buffer has a queue of one allophone, allowing transmission of an allophone while one is being spoken.

When the address latch data line mode of operation is used, set up and hold times must be met. The control lines are manipulated using PIA lines. The critical timing may now be managed using software.

4. SOFTWARE

4.1. MONITOR CONCEPT

4.1.1. KEYBOARD MONITOR

For manual operation, there are two interfaces available to instruct the device, keyboard and microphone. The keyboard monitor must perform the following basic functions:

a) scan the keyboard for an instruction

b) return the instruction if a key is pressed

c) return previous instruction if a key is not pressed

The keyboard is a 4 x 4, 16 button, matrix code configuration. Unlike the single pole/common bus or 2 out of 8 code configurations, the matrix code system does not possess a pin common to all buttons. Instead, a row column orientation is used which requires the fewest interfacing lines. For a 4 x 4 keyboard, 8 interface lines are necessary. The 2 out of 8 code system requires 9 lines while the single pole/common bus requires 17 lines. By using a matrix encoded keyboard, only one PIA port is required.

The keyboard monitor must use the row column data to determine which button was pressed. Figure C-19 illustrates several features of the PIA to keyboard interface. The rows are connected to the four lower order interface lines while the columns occupy the 4 higher

order positions. The PIA pins, associated with the matrix columns, are tied to ground through pull down resistors.

The keyboard monitor will set a row and test the columns (see figure 4-01). If a column is at a logic "1", a button was pressed. The row and column are saved, in order to identify which key was pressed. The keyboard monitor is now exited and the new instruction is executed.

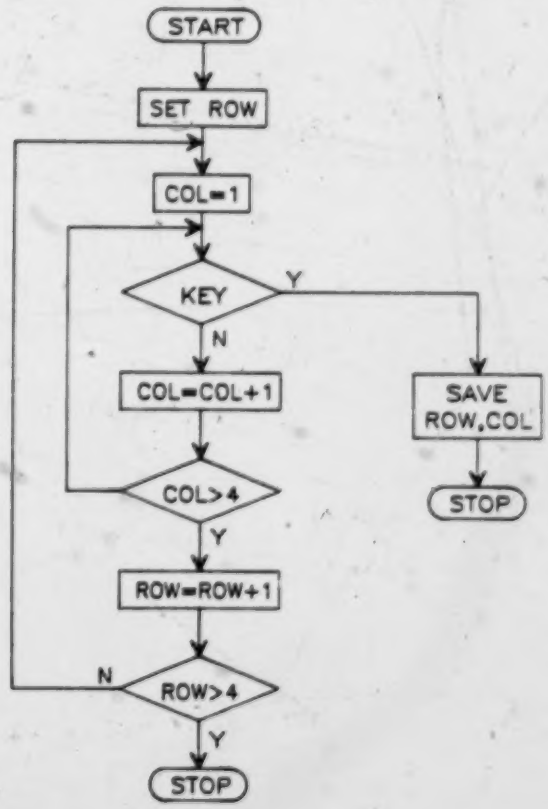


Fig. 4-1: KEYBOARD MONITOR FLOWCHART

If all of the columns are at a logic "0", no buttons

were pressed. If a key has not been pressed, the next row is set and the columns are retested. The entire keyboard has been checked after all the rows have been set and columns tested. If no button was pressed, the row column data is left unchanged. The keyboard monitor is exited and the previous instruction is executed.

4.1.2. MANUAL MONITOR

The manual monitor controls the manipulator during manual operation and performs two functions:

a) instruction determination

b) instruction execution.

The instruction determination phase may be accomplished using two different methods, keyboard or voice recognition input. First the manual monitor determines the controlling device by observing the state of the KEYBOARD/VOICE switch. Next, the appropriate interface monitor is executed in order to obtain the next instruction (see figure 4-02).

Once a valid instruction has been determined, the appropriate command routine is executed. (Refer to the operator's manual for a complete description of available commands.) If an instruction had not been provided to the interface monitor, the previous one is executed. Finally, the MANUAL/AUTOMATIC switch is tested to determine the desired operating mode.

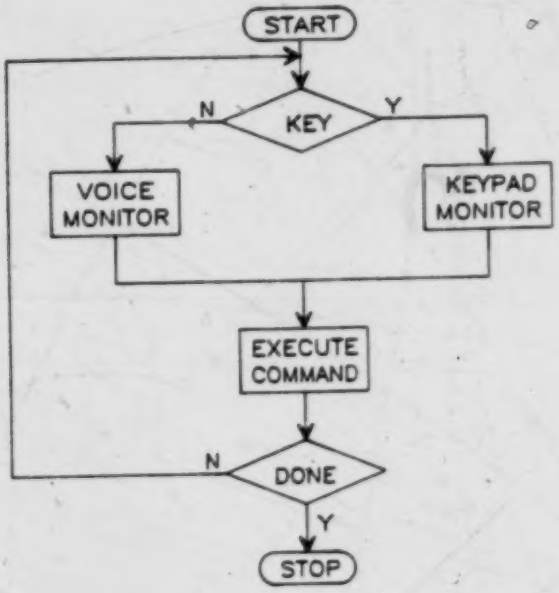


Fig. 4-2: MANUAL MONITOR FLOWCHART

4.1.3. DIAGNOSTICS MONITOR

The diagnostics monitor allows access to a set of subroutines provided for trouble shooting. To enter this monitor, set switch 8, on the DIP switch set located on board C, to the off position. A chosen diagnostic routine is selected via the remaining seven switches. (Refer to the service manual for a complete description of available diagnostic routines.)

The remaining DIP switches are used as a pointer into a look up table, with switch 1 occupying the least

significant position and switch 7 occupying the most significant. Since the look up table contains the addresses of the diagnostic subroutines, the binary switch data must be converted to 16 bit increments. This is performed by shifting the switch data left one bit and clearing the least significant bit.

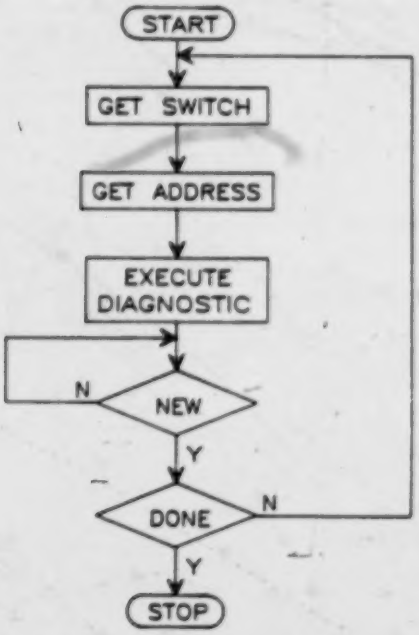


Fig. 4-3: DIAGNOSTICS MONITOR FLOWCHART

The diagnostics monitor is a cyclic process and remains active until the user sets switch 8 to the on position (see figure 4-03). First the switch data is

converted to the look up table pointer. The subroutine address is located and the appropriate diagnostic is performed.

The monitor now waits for a new switch setting. If the test was a repeatable one, it will continue. On the other hand, if the test is not repeatable, the monitor just waits for a new switch setting. Once a new switch configuration is obtained, switch 8 is tested for a completion condition. If the another test has been selected, the entire procedure is repeated, otherwise, the diagnostics are exited.

4.2. AUDIO INTERFACE

4.2.1. ROOM NOISE

Since the incoming audio signal is tested for a rising and falling edge, tp determine the start and finfsh of an utterance, the signal level during periods of silence must be determined. During these periods of silence, the magnitude of the incoming audio signal is referred to as the noise level. The determination of this noise level must be performed prior to any template generation.

There are two possible methods which may be employed to determine the noise level, signal averaging and maximum determination. The first method takes a series of consecutive samples and uses the average as the noise level. This method was tried, using thirty frames, and

found to fail under certain circumstances.

Refer to figure 4-04 for the following discussion. Let the total time, 30T, be equivalent to thirty frames and the signal, s(t), be representative of an audio signal during a period of silence. Note that s(t) often rises above the average, sAVE(t), for various lengths of time. If sAVE(t) is used as the noise level, any point where s(t) > sAVE(t) will be interpreted as a rising edge in the audio signal. This will start the system executing the template generation procedure.

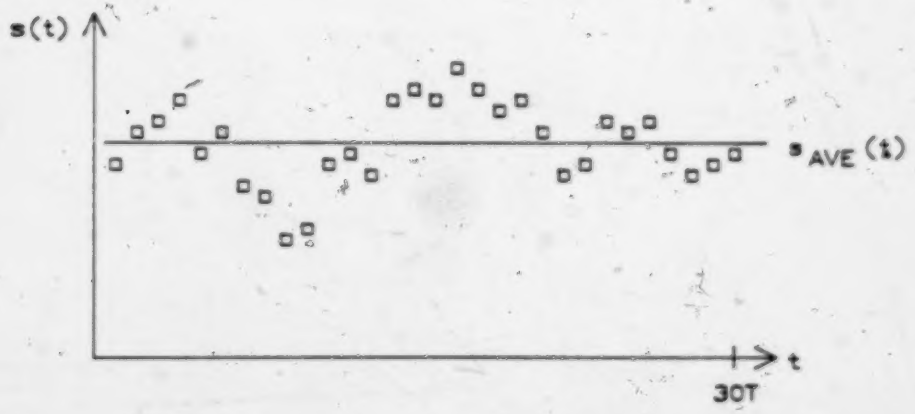


Fig. 4-4: SIGNAL FOR NOISE LEVEL DETERMINATION

These false starts can result in several undesirable occurrences. First, if the noise exists above the average for a significant length of time, the system interprets

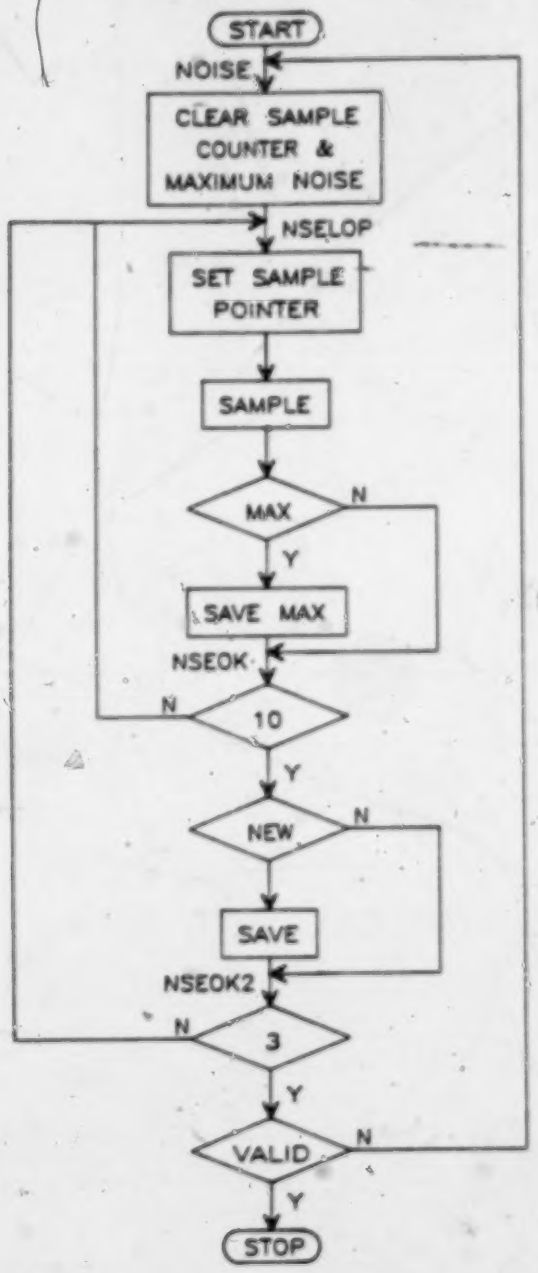


Fig. 4-5: NOISE ROUTINE

the incoming signal as the beginning of an utterance. Another case occurs when the signal falls below the average during an utterance. For this case, above average noise distorts the determination of the falling edge or end of an utterance. Note that the worst case, which occurs quite often, is characterized by noise at both the beginning and end of an utterance.

The maximum noise level method also uses a series of thirty consecutive frames. Instead of averaging the samples, the peak signal level is used as the noise reference level. This produces distinct rising and falling edges for accurate determination of utterance endpoints. This prevents the occurrence of false starts and stops, a series of relatively high noise being interpreted as an utterance, and the retention of extraneous data at the extreme ends of an utterance.

The noise routine uses the latter method to test the incoming audio signal to determine the ambient noise level. Thirty frames are tested to arise at the maximum noise value. The thirty frames are arrived at using two nested loops which take three sets of ten frames (see figure 4-05).

The variables **NCOUNT**, the total sample counter, and **NSEMAX**, the maximum noise level, are initially zeroed. The outer most loop counts the number of ten frame samples and whose start is indicated by the label **NSERPT**. The

initialization of this loop consists of clearing the variable which contains the maximum noise level for the current set of tens frames and initializing the buffer which contains the noise levels.

The inner most loop generates the set of ten samples and determines the maximum noise level for the set. The X index register is initialized with the buffer which saves the coefficients generated during one frame. The SYNC operation uses the interrupt service routine which performs the frame sampling.

After the frame sampling, the noise level is saved in the noise level buffer. The current noise level is tested and if it is the maximum it is saved. If this is not the tenth sample, the inner loop is repeated.

Upon completion of the inner loop, the program falls into the remainder of the outer loop. The ten frame maximum noise level is tested and saved if it is the maximum set noise level. The set counter is tested and if three sets have not been taken, the outer loop is repeated.

After thirty frames have been tested, the maximum noise level is tested for validity. If the maximum noise level is to large, a rising edge may never be detected and template generation will never occur. If the noise level is too high, the whole acquisition process is repeated otherwise the maximum noise level determination is

finished.

4.2.2. AUDIO INPUT

The SP1000 interrupts the system every 10 milliseconds to produce the analysis frames. An analysis frame is 10 milliseconds in duration and the SP1000 returns the average of the signal during this time. The coefficients supplied from the SP1000 are updated after an interrupt; therefore the coefficients only have to be read once per frame. Since coefficient updates are derived from interrupts, frame sampling is a interrupt service routine.

Initially, the counter used to monitor the number of samples is set to zero. The SP1000 is initialized in preparation for the coefficient sampling.

During the SP1000 initialization, the parameter address register was loaded with the value to point at the incoming audio energy. In order to read a parameter, first an initiate fetch operation must be performed followed by reading the parameter data output register. After the ENERGY has been read, the timer which produces the interrupts is reinitialized.

The input testing routine is a cyclic process which will monitor a maximum of ten consecutive routines. Each frame is tested against the value computed by the NOISE routine. Since the start of the prompt is indicated by a rising edge on the incoming audio signal, the loop continues until either ten frames have been tested or the

signal rises above the noise level.

4.2.3. SPEECH SYNTHESIS

The **TALK** subroutine is called to send the speech processor the appropriate allophones, required for a particular word. Indexed addressing is used to retrieve the allophones. The system operates in a closed loop until the word terminator, $FF, is retrieved.

At the end of an utterance, a pause is sent to make the speech processor stop talking the last allophone. To further insure silence, during periods of inactivity, the speech processor's reset line is manually controlled with software. At the end of a word, the reset line is pulled low, thus disabling the speech processor. Therefore, upon entry into the **TALK** routine, the speech processor must be enabled by pulling the reset line high.

The speech synthesizing software uses two interface lines for the loading of allophones. Prior to sending an allophone, the load request line must be tested to insure that the speech processor is ready to accept data. After sending an allophone, it must be latched into the speech processor. This is accomplished by strobing the address latch data line low.

4.3. MANUAL MOVEMENT

For manual operation, each instruction performs different functions. Although only the "move right" instruction will be discussed, the basic concepts may be

extended to the other commands. Figure 4-6 illustrates the sequence of events required to move the manipulator to the right.

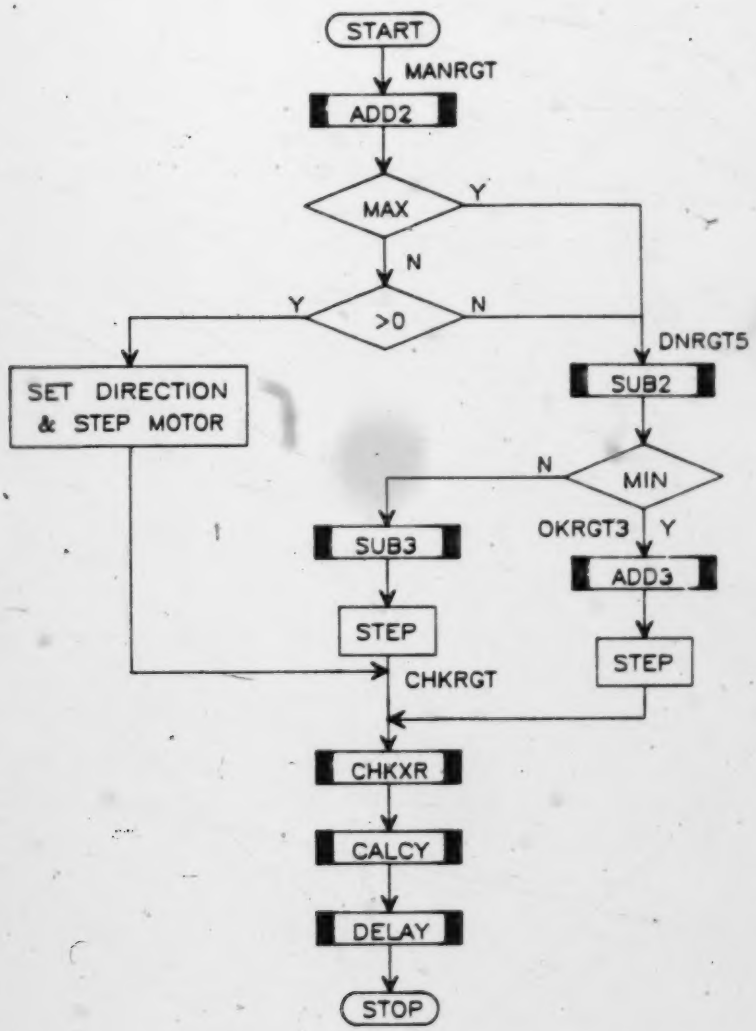


Fig. 4-6: MOVE RIGHT FLOWCHART

Movement to the right only requires stepper motors 2 and 3. For all commands only one motor is used at a time, ie. before the second motor is employed, the first motor is moved to its limit. For this case stepper motor 2 is moved first and an optoswitch is used as the limit. Motor 2 was selected first because it is capable of generating more torque than motor 3.

Upon entry, the optoswitch, associated with motor 2 is tested. If the optoswitch corresponding the maximum angle, the direction for motor 2 is set and it is moved. This continues until angle 2 reaches its maximum, at this time motor 3 is used.

Motor 3 is not limited by an optoswitch, but instead uses a predefined angle. Motor angle 3 is tested to determine if it is greater than or less than the predefined angle. This information is used to set the direction of rotation and the motor is moved.

At this point, **CHKRGT**, the two motor operations rejoin and the remainder of the routine is always performed. There are three remaining functions to be performed. First, the orientation of the manipulator tip, in the x coordinate direction, must be corrected. This process was explained in the design considerations. Since the tip is being moved to the right, the y coordinate is updated. These two previous procedures insure accurate coordinate monitoring in the x-y plane. Finally, speed

control is accomplished by executing a variable delay.

5. RESULTS

5.1. DISTANCE ERRORS

An experiment was performed in order to determine the distance error produced by various masses. First the manipulator was programmed using the teaching phase. The final point was noted and is used as the target for the remaining tests. Various masses were suspended from the manipulator tip and the repeat instruction was used to move the manipulator through the preprogrammed path to the final destination. The distance error is the measured distance from the manipulator tip to the desired endpoint. The experiment is performed through a varying range of masses and repeated for the four different possible speed settings. The test is summarized in table 5-1.

WEIGHT (gms)	SPEED1	SPEED2	SPEED3	SPEED4
60	7	5	4	4
120	7	6	5	5
130	9	7	6	6
140	10	9	7	8
150	16	13	11	9
160	17	18	17	18
170	19	19	19	16
180	20	19	18	18
210	28	28	28	28
240	30	30	30	30

Table 5-1: WEIGHT (gms) VS DISTANCE ERROR (mms)

In analyzing table 5-1, it is important to note that "speed1" corresponds to the fastest speed while "speed4"

is the slowest speed. The data may be more readily analyzed if it is first converted into a percentage format. This is achieved using equation 5-1.

$$\text{percent error} = \frac{\text{distance error}}{\text{length}} * 100 \qquad (5\text{-}1)$$

In equation 5-1, length is the total distance travelled, 20.5 cm. The distance error is the difference between the final endpoint and the desired endpoint. These values are located in table 5-1. The percent distance error is summarized in table 5-2.

WEIGHT (gms)	SPEED1	SPEED2	SPEED3	SPEED4
60	3.4	2.4	2.0	2.0
120	3.4	2.9	2.4	2.4
130	4.4	3.4	2.9	2.9
140	4.9	4.4	3.4	3.9
150	7.8	6.3	5.7	4.4
160	8.3	8.8	8.3	8.8
170	9.7	9.7	9.3	7.8
180	9.8	9.7	8.8	8.8
210	13.7	13.7	13.7	13.7
240	14.6	14.6	14.6	14.6

Table 5-2: WEIGHT (gms) VS DISTANCE ERROR (%)

For the lower loads, the distance error increases with manipulator speed. As the load increases, the distance errors begin to converge for the different manipulator speeds. Finally, at the higher loads, the distance error is independent of manipulator speed. This is the expected result since both loading and rotation

speed produce more slippage in the stepper motors.

5.2. DOWNWARD FORCE

The downward force exerted was measured for the four different speed settings. The manipulator was operated in manual mode using the down instruction. The manipulator tip force was measured by lowering the tip on a scale. The results are recorded in table 5-3.

	SPEED1	SPEED2	SPEED3	SPEED4
Weight(lbs)	2.5	2.5	2.5	2.5

Table 5-3: MAXIMUM DOWNWARD FORCE (lbs)

As seen in table 5-3, the maximum downward force is independent of manipulator speed.

6. FUTURE DEVELOPMENTS

6.1. SPEECH RECOGNITION

Speech recognition uses dynamic time warping and time normalization to compare the features between two templates. A template is the summary of the prominent features of an incoming audio signal. The features retained represent the largest transitions of the incoming signal. The template is comprised of elements which contain periods of change, for example a change in inflection or articulation.

The dynamic time warping procedure compares the elements of two templates, an unknown and a reference. The resulting distance measure represents the relative closeness between the two templates. Thus, the lower the distance measure the closer the match and vice versa for large distance measures.

Time normalization maybe used to improve upon the dynamic time warping technique. This procedure attempts to remove any time distortions by retaining only the maximum changes in the rate of speech and rhythm variations. Time normalization compresses periods when the signal is relatively constant and ignores periods of transition.

6.1.1. DYNAMIC TIME WARPING

Distance measure evaluates the closeness between two templates. The smaller the distance measure, the closer the match. Distance measure is performed on a frame to

frame basis.

One method of computing the distance measure is the sum of the squares method:

$$\underline{Q}(i,j) = [\underline{V}(i,j) - \underline{H}(i,j)]2 \qquad (6\text{-}1)$$

Note that the following definitions are maintained throughout the course of the discussion :

1) $\underline{Q}$ - distance measure

2) $\underline{H}$ - reference value

3) $\underline{V}$ - unknown value

For one frame, the distance measure is the sum of the squares of one of the differences for the eight LPC coefficients. An alternative method is the Chebyshev distance measure procedure :

$$\underline{Q} = \text{ABS } [\underline{V}(i,j) - \underline{H}(i,j)] \qquad (6\text{-}2)$$

Calculating the Chebyshev distance measure, for one frame, is more efficient due to the absence of the squaring operation.

---

i) 5 4 3 2 1	i) 5 4 3 2
ii) 4 5 5 5 4	ii) 4 5 4 3
iii) 1 3 3 2 5	iii) 2 3 5 5
a) Reference	b) Unknown

Fig. 6-1: EXAMPLE TEMPLATES

The distance matrix, $\underline{Q}$, contains the distance measures between all the frames for an unknown and

reference template. As an example, compare a five frame, three coefficient reference template to a four frame, three coefficient unknown template (see figure 6-1).

For this example, the Chebyshev distance measure is calculated as follows :

$$\begin{aligned} \underline{Q}(i,j) = & \text{ ABS}[\underline{H}i(i,j) - \underline{V}i(i,j)] \\ & + \text{ ABS}[\underline{H}ii(i,j) - \underline{V}ii(i,j)] \\ & + \text{ ABS}[\underline{H}iii(i,j) - \underline{V}iii(i,j)] \end{aligned} \qquad (6\text{-}3)$$

Consider the first element, its distance measure is calculated as follows :

$$\begin{aligned} \underline{Q}(1,1) = & \text{ ABS}[\underline{H}i(1,1) - \underline{V}i(1,1)] \\ & + \text{ ABS}[\underline{H}ii(1,1) - \underline{V}ii(1,1)] \\ & + \text{ ABS}[\underline{H}iii(1,1) - \underline{V}iii(1,1)] \\ = & \text{ ABS}[5-5] + \text{ABS}[4-4] + \text{ABS}[1-2] \\ = & \ 1 \end{aligned} \qquad (6\text{-}4)$$

The distance measure of "1" indicates that the two frames compare very closely. As the distance measure increases, the frame similarity decreases.

The remaining distance measures are calculated and the distance matrix, $\underline{Q}$, is built (see figure 6-2). Note that the element for row 2, column 2 indicates an exact match and the distance measure at row 4, column 1 indicates a poor match.

Dynamic time warping provides nonlinear time alignment for utterance comparisons. The need for time alignment arises from the variations in duration and

volume when a word is spoken. Dynamic time warping expands and contracts the template to align the frames, in an attempt to match two utterances.

---

	1 4 5	3 5 4	3 5 3	2 5 2	5 4 1
2 4 5	1	3	4	4	8
3 5 4	4	0	1	3	6
5 4 3	6	4	3	5	2
5 3 2	8	5	5	5	2

Fig. 6-2: DISTANCE MEASURE MATRIX

Dynamic time warping produces an optimal solution, distance measure, to a global problem, a template. This is a sequential process that utilizes local decisions to form the global solution. The template distance measure is developed by using the already known frame distance measures.

This algorithm develops a dynamic programming matrix which contains a cumulative score of the frame distance measures. An individual local distance measure is calculated as follows :

$$\underline{S}(i,j) = \underline{Q}(i,j) + MIN[\underline{S}(i-1,j), \underline{S}(i-1,j-1), \underline{S}(i,j-1)] \quad (6\text{-}5)$$

The local decision process is illustrated in figure 6-3.

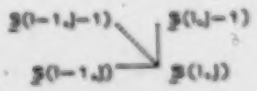


Fig. 6-3: LOCAL DECISION

Since the goal is to minimize the overall distance measure, the minimum from the previous calculations is added to the current frame's distance measure.

The local decision process is a three way function. The point in question, $\underline{S}(i,j)$, can be arrived at along three different paths; horizontally, vertically or diagonally. To minimize cost, at the current point, the minimal cost of the three previous points must be considered. To minimize the current distance measure, add the minimum distance measure from the previous points to the distance measure which accounts for the current point, $\underline{Q}(i,j)$.

Using the distance matrix $\underline{Q}$, from the previous example, the cumulative measure at row 2, column 2 is calculated as follows :

$$\underline{S}(2,2) = \underline{Q}(2,2) + \text{MIN}\{\underline{S}(1,2),\underline{S}(1,1),\underline{S}(2,1)\}$$
$$= 0 + \text{MIN}\{3,1,4\}$$
$$= 1 \qquad (6\text{-}6)$$

Continuing on, the dynamic programming matrix, $\underline{S}$, can be developed (see figure 6-4).

	1 4 5	3 5 4	3 5 3	2 5 2	5 4 1
2 4 5	1	4	7	11	19
3 5 4	5	1	2	5	11
5 4 3	11	5	4	7	7
5 3 2	19	10	9	9	9

Fig. 6-4: DYNAMIC PROGRAMMING MATRIX

Recall that the goal was to compute the cumulative distance measure for a template. In terms of the dynamic programming matrix, a cost effective path from the top left to the bottom right corner must be found. Note that the top left corner references the beginning of the

templates while the bottom right represents the end.

The distance measure, in the bottom right corner, represents the global distance measure for the comparison of the two utterances. This result is the optimized solution using the individual frame distance measures in a sequential manner. Like the frame distance measures, the magnitude of the global distance implies a degree of closeness. In this example, a final measure of "9" was arrived at, indicating a reasonably close match.

6.1.2. TIME NORMALIZATION

Dynamic programming techniques limit the acceptable range of time distortions. A single pass time normalization procedure is used to aid the dynamic programming algorithm. The distance measure calculations are simplified by reducing the number of frames.

Time distortion is primarily the result of changes in the speech rate and rhythm variations. During steady state portions of the speech signal, both speech rate and rhythm remain relatively constant. This will produce a series of frames with similar characteristics.

Time normalization attempts to reduce the data by extracting and saving only the unique features of the speech signal. The unique features occur only during periods of transition, not steady state. Two concurrent processes produce the resulting time normalized representation of the speech signal. First, successive

similar frames, from a steady state signal, are represented by a single frame. Second, only data from periods of transition are retained.

The time normalization algorithm is referred to as time compression. Time compression is subdivided ito two methods, fixed and variable length. Fixed length time compression will reduce the unknown and known templates to the same length. Although the intrinsic length of the template is lost, retention of the largest transition levels is sufficient for isolated word recognition.

One method of time compression is the fixed length trace segmentation method. Let "p" indicate the number of features retained at a sampling instant. For p features, at a sampling instant, let the incoming speech signal be described by :

$$\underline{S}(t) = \{S1(t), S2(t), S3(t), \ldots, SP(t)\} \qquad (6\text{-}7)$$

For fixed length trace segmentation, the speech signal is approximated through p dimensional space. The approximation uses lengths and rhythms in the feature space. The time normalization is achieved through the following transformation :

$$\underline{S}N(w) = S[\phi S(w)] \qquad (6\text{-}8)$$

To achieve time compression, first calculate the total length of the utterance. The N frames retained are extracted at N regularly spaced points on the utterance.

This method has two unacceptable characteristics.

There is no guarantee that steady state frames are compressed into one. There also exists the possibility that the speech signal is at steady state along the N equally spaced points. By selecting equally spaced points, the periods of maximum transition may be missed.

An alternative approach is to use the transition frame selection technique. This method is based on the spectral instability function :

$$\underline{INST}(t) = || \underline{S}(t) - \underline{S}(t-T) || \qquad (6\text{-}9)$$

The speech signal, $\underline{S}(t)$, is as previously defined and T is a sampling period. The spectral instability function calculates the difference between the current sample and the previous sample.

The relative minima indicate stable (steady state) portions of the speech signal. Periods of transition are represented by the relative maxima. The larger the relative maxima, the greater the difference during the transition level. The smaller the relative minima, the greater the stability of steady state regions. Time compression is accomplished by retaining the largest maxima, the regions of greatest transition. The spectral instability function satisfies the two criteria stated for feature extraction time compression, steady state compression and maximum transition retention.

Now consider the following example with three features per sample (see figure 6-5).

---

i) 5 4 2 2 1 5 2 3 2

ii) 4 5 5 5 4 4 5 4 3

iii) 1 3 3 2 5 2 3 4 5

Fig. 6-5: EXAMPLE UTTERANCE

After the spectral instability function is applied, the following spectrum is acquired (see figure 6-6).

---

i) 1 1 2 0 4 3 1 1

ii) 1 0 0 1 0 1 0 1

iii) 2 0 1 3 3 1 1 1

Fig. 6-6: DIFFERENCE UTTERANCE

For four feature retention, the following features are saved for dynamic programming (see figure 6-7).

---

i) 1 2 4 3

ii) 1 1 1 1

iii) 2 1 3 3

Fig. 6-7: COMPRESSED UTTERANCE

6.2. CLOSED LOOP CONTROL

6.2.1. PURPOSE

Closed loop control may be added to achieve step detection. During programmed operation, missing steps will

produce an error in the manipulator path. During manual operation, stepping errors will result in the system calculating inaccurate coordinates.

Step detection will be used to determine whenever the desired step does not occur. Should this happen, the system will continue to attempt to move the motor. This continues until a step is detected. By continually monitoring for steps, the manipulator should never deviate from a predescribed or maintain incorrect manipulator coordinates.

The most commonly used closed loop control is provided through the use of optical encoders (see Kenjo). Optical encoders posses three primary limitations :

1) expensive, sometimes exceeding the price of the stepper motor
2) must be manually adjusted relative to the rotor's reference point, often difficult to perform
3) dependent upon the enviroment, easily influenced by temperature and electrical noise.

6.2.2. CURRENT SENSING

Instead, rotor position may be monitored through the use of the current waveform. This removes the necessity of converting mechanical positions into electrical signals and the coupling of a mechanical encoder to the rotor. Figure 6-8 illustrates typical current waveforms in the stepper motor windings.

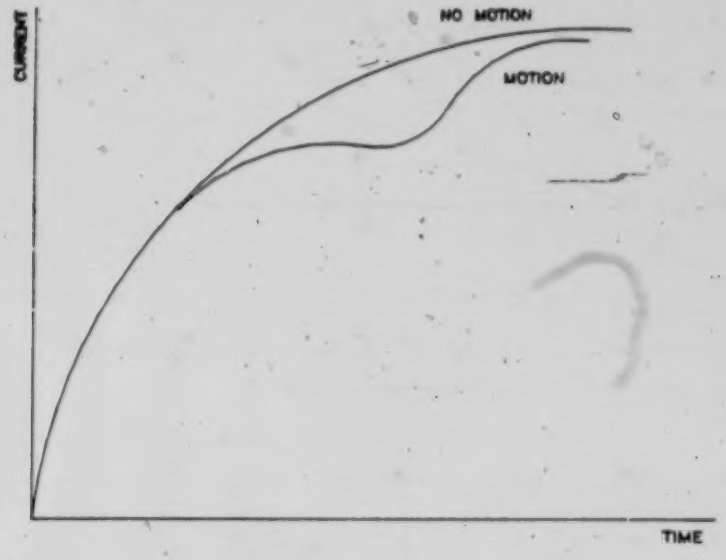


Fig. 6-8: CURRENT WAVEFORMS

There exists an obvious difference between the motion and no motion waveforms. This difference may be used to achieve closed loop control. Motion is detected by monitoring voltage crossings from a comparator which receives the winding signal and an exponential waveform.

7. CONCLUSION

This robotic manipulator is designed with two specific objectives. First the cost of the design and construction of this device must be minimized. This allows the largest number of handicapped people the opportunity to use such a device.

Secondly, the device must be capable of fulfilling the requirements of the individual. The instruction modes are designed to configure the device to the individual and their corresponding requirements. Thus, the same device may be applied to many different users and handicaps.

The robotic arm will allow handicaped patients the opportunity to feed themselves. The addition of voice interaction and a keypad will increase the versatility of the device. In addition to the feeding mode, this allows the user to manually control the manipulator. For the feeding operation, an instruction sequence teaches the manipulator the required motion.

8. REFERENCES

1 **Acarnley, P. P.**, "Stepping Motors : A Guide to Modern Theory and Practice", P. Peregrinus on Behalf of IEEE, Stevenage (England), 1982, TK2511 Science

2 **Anton, Howard** and **Korres, Chris**, "Applications of Linear Algebra", New York, John Wiley and Sons, 1977

3 **Apple, H. P.** and **Reswick, J. B.**, "A Multi-Level Approach to Orthotic/Prosthetic Control System Design", Advances in External Control of Human Extremities, Belgrade, Yugoslavia, 1970, pp 323 - 338

4 **Corker, Kevin, Lyman, John H.** and **Sheredos, Saleem**, "A Preliminary Evaluation of Remote Medical Manipulators", Bulletin of Prosthetics Research, Fall 1979, pp 107 - 134

5 **General Instrument Corporation, Microelectronics Division**, "SP1000 Voice Recognition/Synthesis Device, Preliminary Information", 600 West John Street, Hicksville, New York, 11802

6 **Gibbons, David T., O'Riain, Micheal D.**, and **Reuben, Keith A.**, "A Microprocessor Controlled Arm Prosthesis", Northeast Bioengineering Conference, pp 106 - 109

7 **Guittet, Jacques, Kwee, Him Hok, Quetin, Nicole** and **Yelon, Joel**, "The Spartacus Telethtesis: Manipulator Control Studies", Bulletin of Prosthesis Research, Fall 1979, pp 69 - 105

8 **Hawley, Mones E.**, "Speech Intelligibility and Speaker Recognition", Stroudsburg, Pennsylvania, Dowden, Hutchinson & Ross Inc., 1977 TK 7882 Science

9 **Hunt, M. J.**, "Speaker Differences in Speech and Speaker Recognition", North Atlantic Treaty Organization Advisory Group For Aerospace Research and Developement, AGARD Lecture Series NO.129, Speech Processing, TK 7882, Science

10 **Kenjo, Takashi**, "Stepping motors and their microprocessor controls", Oxford, Oxford University Press, 1984, TK2785 K33 Science

11 **Kuo, Benjamin C.**, "Theory and Applications of Step Motors", West Publishing Co., New York, 1974, TK2785 S98 Science

12 **Leifer, Larry, Sachs, James** and **Sun, Rickson**, "A Robotic Aid for the Severely Disabled - Microprocessor Controls", Joint Automatic Conference, 1979, pp 777 - 789

13 **Leininger, Steven W.**, "Colne Robotics Android, The Small Systems Robot", Byte, May 1982, pp 286 - 294

14 **Leventhal, Lance A.**, "6809 Assembly Language Programming", Osbourne/McGraw-Hill, Berkeley, 1981

15 **Linggard, R.**, "Electronic Synthesis of Speech", Cambridge, Cambridge University Press, 1985, TK 7882 Science

16 **May, Janet**, "Microelectronics Application Report : Allophone Speech Synthesis Technique", General Instrument, 1982

17 **Moore, R. K.**, "Techniques for Automatic Speech Recognition", North Atlantic Treaty Organization Advisory Group For AeroSpace Research and Developement, AGARD Lecture Series NO.129, Speech Processing, TK 7882, Science

18 **Moore, R. K.**, "Assessment of Speech Systems", North Atlantic Treaty Organization Advisory Group For AeroSpace Research and Developement, AGARD Lecture Series NO.129, Speech Processing, TK 7882, Science

19 **Morgan, Nelson**, "Talking Chips", New York, McGraw Hill Book Company, 1984, TK 7882, Science

20 **Mostek Corporation**, "Mostek Bytewyde Memory Data Book", 1215 W. Crosby Road, Carrollton, Texas, 75006, 1981

21 **Motorla Semiconductor Products Inc.**, "MC6839 Floating Point ROM User's Manual", 3501 Ed Bluestein Blvd., Austin, Texas, 78721, October 1981

22 **Motorola Semiconductor Products Inc.**, "MC6809 - MC6809E Microprocessor Programming Manual", 3501 Ed Bluestein Blvd, Austin, Texas, 78721, 1981

23 **Motorola Semiconductor Products Inc.**, "Motorola Microprocessor Data Manuals", 3501 Ed Bluestein Blvd., Austin, Texas, 78721

24 **Motorola Semiconductor Products Inc.**, "MC6881 Floating-Point Coprocessor User's Manual", 3501 Ed Bluestein Blvd., Austin, Texas, 78721

25 **National Semiconductor Corporation**, "MM57109 MOS/LSI Digit-oriented Microprocessor", 2900 Semiconductor Drive, Santa Clara, California, 95051, January 1977

26 **Oja, Dr. E.**, "Subspace Methods of Pattern Recognition", Letchworth, England, Research Studies Press, 1983, TK 7882, Science

27 **Ramey, R. L., Aylor, J. H.** and **Williams, R. D.**, "Microcomputer-Aided Eating for the Severly Handicapped", Computer, January 1979, pp 54 - 61

28 **Roesler, H., Kuppers, H. J.** and **Schmalenbach, E.**, "The Medical Manipulator and its Adapted Enviroment: A System for the Rehabilitation of Severely Handicapped", Proceedings of the First International Conference on Telemanipulators for the Physically Handicapped, Rocquencourt, France, 1978

29 **Roth, Charles H. Jr.**, "Fundamentals of Logic Design", Second edition, St. Paul, West Publishing Company, 1979

30 **Schmeisser, Gerhard** and **Seamone, Woodrow**, "Low Cost Assistive Device Systems for a High Spinal-Cord-Injured Person in the Home Enviroment - A Technical Note", Bulletin of Prosthetics Research, Fall 1979, pp 212 - 223

31 **Schneider, Wolfger, Seamone, Woodrow** and **Schmeisser, Gerhard**, "A Microprocessor-Controlled Robotic Arm Allows Self-Feeding for a Quadriplegic", Proceedings of the IEEE Computer Society Workshop on Application of Personal Computing aid to the Habdicapped, April 1980

32 **Schneider, Wolfger, Schmeisser, Gerhard** and **Seamone, Woodrow**, "A Computer-Aided Robotic Arm/Worktable System for the High-Level Quadriplegic", Computer, January 1981, pp 41 - 47

33 **Scott, Victoria**, "Life Enhancement for the Handicapped Through Voice Technology", Speech Technology, April 1982, Volume 1, No. 2, pp 63 - 65, TK7882 S65 Science

34 **Silicon Systems**, "User's Guide for Phonetic Programming Using the SSI 263A", 14351 Myford Road, Tustin, California, 92680, 1984

35 **Silicon Systems**, "SSI 263A Phoneme Speech Synthesizer Data Sheet", 14351 Myford Road, Tustin, California, 92680, 1985

36 **Teja, Edward R.** and **Gonnella, Gary W.**, "Voice Technology", Reston, Virginia, Reston Publishing Company Inc., 1983, TK 7882 Science

37 **Williams, John M.**, "CASH is Providing Careers for Disabled People", Speech Technology, February/March 1985, Volume 2, No. 4, pp 30 - 32, TK7882 S65 Science

38 **Veltri, Steven J.**, "How to Make Your Computer Talk", New York, McGraw Hill Book Company, 1985

39 **Weissberger, Alan J.** and **Toal, Ted**, "Tough Mathematical Tasks are Child's Play for Number Cruncher", Electronics, February, 1977, pp. 102 - 107

A. SERVICE MANUAL

A.1. DIAGNOSTIC TEST CONCEPT

The diagnostic tests are subdivided into two classes, "debug" and "probe", for hardware testing. The diagnostic "debug" tests utilize test subroutines which reside in ROM. The "probe" tests involve observing measurements and waveforms for particular events and components.

To enter the diagnostic "debug" tests, set switch 8, of the switch package located on board C, to the "off" position. The remaining seven switches are used to select the desired test. The diagnostic display values, listed at the end of this section, summarize the "debug" tests. To select a test, set the switches to the binary value of the test number with switch 1 being the LSB. For example, to select test 5 set switch 3 and 1 to the on position. Note that, unless specified, the test number will be shown on the diagnostic display.

A.2. DIAGNOSTIC DEBUG TESTS

A.2.1. DIAGNOSTIC DISPLAY TEST

The diagnostic display routine will test the seven segment displays by counting and displaying the binary coded decimal (BCD) values from 0 to 99. A failure in this test may be caused by several different factors.

If there is a problem with one or more of the segments, check the current sinking resistors. The segment

problems will usually be caused by a open circuit condition in the path of the particular segment. If the display does not count in the correct sequence, the problem probably lies within the BCD to seven segment decoding circuitry. Finally, if the display does not count or show anything the problem is associated with the third PIA, I1, on board C.

A.2.2. LED TESTS 1 - 2

The light emitting diode (LED) test turns all of the LEDs on and off at one half second intervals. This test should be performed whenever a switch test fails, in order to insure that the LEDs are operational. LED test 1 operates on the LEDs dedicated to the "STANDBY" and "LEARN/OPERATE" switches. The second LED test verifies the LEDs associated with the "VOICE/KEYPAD" and "MANUAL/AUTOMATIC" switches.

PIA	PIN#	LED	PIA	PIN#	LED
H1	3	LEARN	K1	11	AUTOMATIC
H1	6	STANDBY	K1	12	VOICE
H1	9	OPERATE	K1	13	MANUAL
K1	10	KEYPAD			

Table A-1: LED-PIA CONNECTIONS

Should either test fail, use the table above to check the interface pins on PIAs, H1 and K1. If the interface pins are correct, test the cathodes for the

signal which will turn the LEDs "on" and "off". If the cathodes are correct, measure the anodes to make sure they are at 5 Volts. Finally, if all of the preceding tests fail, replace the LEDs. For faster testing, note that if all of the LEDs fail to light, the problem exists in either the PIA or the power supplied to this common anode configuration.

A.2.3. PERIPHERAL INTERFACE ADAPTERS

A.2.3.1. PIA TESTS 1 - 3

These procedures will test three different PIA ports. Of the several tests performed on these interface lines, this particular procedure scrolls a set bit through the port. These test operate as an endless loop until another test is selected.

Initially, the least significant bit is set while the seven higher order bits are cleared. After a short delay, the next most significant bit is set while the seven remaining bits are cleared. Refer to TABLE A-2, located in PIA tests 4 - 6 section, to identify the component pins or the edge connector, which must be monitored.

A.2.3.2. PIA TESTS 4 - 6

Of the several tests performed on these interface lines, this particular procedure scrolls a cleared bit through the port. These tests perform the complement of PIA test number one to three.

Initially, the least significant bit is cleared while the seven higher order bits are at a logic 1. After a short delay, the next most significant bit is cleared while the seven remaining bits are set. Refer TABLE A-2 to identify the component or edge connector pin, which must be monitored.

PIA#	BIT#	PIN#	EC#	PIA#	BIT#	PIN#	EC#
1	0	2	26	1	4	14	9
1	1	3	27	1	5	15	NC
1	2	4	28	1	6	16	NC
1	3	5	29	1	7	17	NC
1	4	6	30	2	0	10	31
1	5	7	NC	2	1	11	32
1	6	8	NC	2	2	12	33
1	7	9	NC	2	3	13	34
1	0	10	1	2	4	14	35
1	1	11	3	2	5	15	NC
1	2	12	5	2	6	16	NC
1	3	13	7	2	7	17	NC

Table A-2: PIA-EC CONNECTIONS

A.2.3.3. PIA TESTS 7 - 9

Of the several tests performed on these interface lines, this particular procedure alternately sets the entire port to a logic 1 and then a logic 0.

Initially, the entire port is set to a logic 1 level. After a short delay, the port is set to a low logic level. Refer to TABLE A-2, located in the PIA tests 4 - 6 section, to identify the pins, on either the component or the edge connector, which must be monitored.

A.2.4. PROGRAMMABLE TIMER MODULES

A.2.4.1. PTM TESTS 1 - 3

The programmable timer tests will check the three PTMs. All three timers, for the first two PTMs, are operated in sixteen bit continuous mode and will produce a continuous square wave at the corresponding output. For the third PTM, only two timers are operated in 16 bit

PTM#	TIMER	PIN#	PTM#	TIMER	PIN#
1	1	27	2	2	3
1	2	3	2	3	6
1	3	6	3	2	3
2	1	27	3	3	6

Table A-3: PTM TIMER OUTPUTS

continuous mode. The third timer output is connected to the non-maskable interrupt line for the tracing function in the monitor. TABLE A-3 provides a quick reference for the timer testing.

A.2.4.2. PTM RESET TEST

This procedure tests the reset for PTMs 1 and 2. These two timers are used by the "LEARN" and "OPERATE" routines to record and playback the robotic manipulator movements. The reset is hard-wired to bit 3 of port A on PIA number 2.

For the PTMs, the reset is active low and can be monitored on pin 8 of components D1 and D2. This test

pulls the reset low and holds it there until the procedure is finished. Prior to starting the test, the reset should be at a logic "1". After the diagnostic switches have been set, the reset should drop to a logic "0". Upon completion of the test, the reset must be once again at a logic "1".

A.2.5. RANDOM ACCESS MEMORY

A.2.5.1. RAM TESTS 1 - 4

When these tests are used, 8 kBytes of random access memory (RAM) for the microprocessor board is tested. First, each byte of memory is cleared and then tested for $00. If the entire block of RAM satisfies these conditions, the test number is alternately flashed with a blank display. Should the test fail, the test number is alternately displayed with a pair of zeros. Replace the memory semiconductor and retest. Refer TABLE A-4 to determine corresponding components and test numbers.

IC#	TEST#	IC#	TEST#
I2	1	I2	5
I3	2	I3	6
B2	3	B2	7
E4	4	E4	8

Table A-4: RAM TEST ICs

A.2.5.2. RAM TESTS 5 - 8

When these are used, 8 KBytes of random access memory (RAM) for the microprocessor board is tested. These

tests perform the compiement of RAM tests 1 - 4. The entire 8 K block of memory is set and then tested for $FF. If the entire block of RAM satisfies these conditions, the test number is alternately flashed with a blank display.

A.2.6. SPEECH SYNTHESIS TEST

This test is used to verify the operation of the speech synthesis circuitry. When the diagnostic switches are set, the speech synthesizer will count from one to ten. This is a cyclic process and is repeated until another test is designated.

A.2.7. STEPPER MOTORS

A.2.7.1. STEPPER MOTOR DIRECTION TESTS

These tests will monitor the direction of rotation for the five stepper motors. These tests employs the LEDs used by the "LEARN/OPERATE" switch to indicate the direction of rotation.

To test the direction sensing capabilities for a stepper motor, simply manually rotate the motor. If the upper, green LED illuminates, then the motor has been rotated in the counter clockwise direction. If the lower, red LED illuminates, then the stepper motor has been turned in the clockwise direction.

A.2.7.2. STEPPER MOTOR MOVEMENT TESTS

These tests are used to test the stepper motor driving circuitry for the five motors. When the diagnostic switches are configured accordingly, the stepper motor may

be manually moved by the operator. The "LEARN/OPERATE" switch is used to determine the direction of rotation with the corresponding LEDs indicating the direction.

When the switch is in the upper, "OPERATE" position, the corresponding red LED should illuminate and the stepper motor should rotate in the counter clockwise direction. For the lower, "LEARN" position, the bottom green LED should illuminate and the motor should rotate in the opposing clockwise direction. Completion of both of these test indicates satisfactory operation of the driving circuitry for the stepper motor under investigation.

A.2.7.3. STEPPER MOTOR PULSE DETECTION TESTS

These tests are used to verify the motor pulse detection circuitry required by the stepper motors during the "LEARN" session. The following table, TABLE A-5, is used to determine the required switch setting for a particular motor. Irregardless of the direction of rotation, a pulse

MOTOR#	EDGE#	MOTOR#	EDGE#
1	31	4	34
2	32	5	35
3	33		

Table A-5: STEPPER MOTOR-EC CONNECTIONS

is produced for every step of the motor. By manually rotating the stepper motor, a pulse train should be

observed at the ribbon cable connector. The previous table indicates which ribbon cable connector pin corresponds to which stepper motor. This waveform should occur on both board C and board D.

A.2.8. SWITCHES

A.2.8.1. DISH ROTATION SWITCH TEST

This routine will test the four position dish rotation switch. This switch is directly connected to the four least significant bits of port A in peripheral interface adapter number four. All four poles of the switch are tested and the results are shown on the display.

The test is initiated by setting switches three and one. At this point, all of the light emitting diodes are illuminated to indicate that a display dependent test is being performed. The diagnostic display shows the pole which the switch is currently utilizing. When the switch position is changed, the display should also change. The following table, TABLE A-6, summarizes which pole corresponds to which display.

DISPLAY	PIA BIT	DISPLAY	PIA BIT
00	0	04	2
02	1	08	3

Table A-6: DISH ROTATION SWITCH-PIA CONNECTIONS

If the test fails there are several possible solutions. First, the diagnostic display test should be performed in order to insure that the display is functional. If the display is operational, the probe test should be performed in order to insure that the problem does not exist in the switch or the corresponding pull up resistors. Should the problem still exist, replace the peripheral interface adapter, K1 on board C, and retest.

A.2.8.2. LEARN OPERATE SWITCH TEST

This procedure will test the "LEARN/OPERATE" switch at the peripheral interface adapter. The switch is debounced through a Motorola CMOS hex debouncing package and connected to bit 0, port A of the second peripheral interface adapter. Therefore, a failure in the debouncing circuitry does not indicate a failure of the PIA.

The test is initiated by setting switch three. At this point, the diagnostic display will illuminate the number eight. Now the learn operate switch is being monitored by the system and the light emitting diode corresponding to the switch position is illuminated. Changing the switch position should cause a change on the light emitting diodes. The illuminated light emitting diode must correspond to the switch position, ie. if the switch is set in the lower position, then the bottom diode must be lit.

If the test fails, there are several possible

solutions. First, if the diodes never light, the light emitting diode test should be performed in order to insure that the diodes are operational. If the diodes are operational, then the debouncing circuitry should be tested. If this test is satisfactorily completed, then replace the peripheral interface adapter, H1 on board C, and repeat.

A.2.8.3. MANUAL OPERATION SWITCH TEST

This test is used to verify the manual operation switch. When this switch is set, all of the LEDs should be illuminated and exactly the opposite when the switch is not set. Being able to enter the diagnostic routines indicates the operation of this switch. If the diagnostics can not be entered, replace the reset ROM with a diagnostic ROM configured with the reset vector. Now the manual operation switch may be tested.

A.2.8.4. STANDBY SWITCH TEST

This procedure will test the "STANDBY" switch at the peripheral interface adapter. The switch is debounced through a Motorola CMOS hex debouncing package and connected to bit 2, port A of the second peripheral interface adapter. Therefore, a failure in the switch test may indicate a failure in either the peripheral interface adapter or the debouncing circuitry.

The test is initiated by setting switch 3 and 0. At this point, the diagnostic display will illuminate the

number nine. Now the standby switch is being monitored by the system and if the switch is set in the upper position, the light emitting diode above it will be illuminated. When the switch is set in the lower position, the two light emitting diodes corresponding to the learn operate switch will be illuminated.

If the test fails, there are several possible solutions. First, if the diodes never illuminate, the light emitting diode test should be performed in order to insure that the diodes are operational. If the diodes are operational, then the debouncing circuitry should be tested. If this test is satisfactorily completed, then replace the peripheral interface adapter, H1 on board C, and repeat the testing procedure.

A.2.8.5. WAIT SWITCH TEST

This routine will test the four position "WAIT" switch. This switch is directly connected to the four most significant bits of port A in peripheral interface adapter number four. All four poles of the switch are tested and the results are shown on the display.

The test is initiated by setting switches three, one and zero. At this point, all of the light emitting diodes are illuminated to indicate that a display dependent test is being performed. The diagnostic display shows the pole which the switch is currently utilizing. When the switch position is changed, the display should also change. Refer

to TABLE A-7 to determine which pole corresponds to which display.

DISPLAY	PIA BIT	DISPLAY	PIA BIT
00	3	04	5
02	4	08	6

Table A-7: WAIT SWITCH-EC CONNECTIONS

If the test fails there are several possible solutions. First, the diagnostic display test should be performed in order to insure that the display is functional. If the display is operational, the probe test should be performed in order to insure that the problem does not exist in the switch or the corresponding pull up resistors. Should the problem still exist, replace the peripheral interface adapter, K1 on board C, and repeat the testing procedure.

A.3. DIAGNOSTIC PROBE TESTS

A.3.1. OSCILLATORS

A.3.1.1. BAUD RATE TEST

The ACIA's are clocked by a selectable baud rate generator. Since the ASSIST09 monitor divides the baud rate by 16, a frequency of 153.6 KHz is required to operate at 9600 baud and should be measured at pin 3 and 4 on the ACIAs. If the clock is absent, the problem resides in either the baud rate generator or between the baud rate

generator and the ACIAs.

If the measured frequency on pin 3, of component A2, is correct, the baud rate generator is operational and the problem resides elsewhere. If a different frequency is observed at the ACIAs, the switches are incorrectly set. Make sure that only switch 5 is set. If no frequency is observed at the ACIAs, a discontinuity exists between the baud rate generator and the ACIAs.

There are three tests which assist in locating the problem. If the square wave is absent on pin 5, of the switch package, the problem lies between the switch and the baud rate generator. However, if the frequency is absent on pin 12, of the switch package, the switch is either not set or broken. For the third case, both sides of the switch are correct and therefore the discontinuity lies between the switch and the ACIAs.

The absence of a waveform on pin 3, of A2, indicates that the baud rate generator is not operating. First, perform the diagnostic probe baud rate generator crystal test. Next, test the reset, pin 10, to insure it is high for a high to a low transition when the system is reset. Both pins 24 and 22 must be at a logic 1 while pins 23 and 12 must be at a logic 0. If all of the previous tests fail, replace the baud rate generator.

A.3.1.2. BAUD RATE GENERATOR CRYSTAL TEST

The baud rate generator is clocked by a 1.84 MHz

crystal. Pin twenty one on A2 should show this frequency on a scope while the crystal out pin, 20, should be at ground. Should these waveforms be absent, replace the resistor and retest. If the test still fails, replace the crystal and retest. If the waveforms are still absent replace the baud generator, A2, and retest.

A.3.1.3. MICROPROCESSOR CRYSTAL TEST

The microprocessor is clocked by a 3.579 MHz crystal. Pin 38, on F2 should show this frequency on a scope while the crystal out pin, 39, should be at ground. Should these waveforms be absent, replace the filtering capacitors and retest. If this test fails, replace the crystal and retest. If the waveforms are still absent, replace the microprocessor and retest.

A.3.1.4. REAL TIME CLOCK CRYSTAL TEST

The real time clock is clocked by a 32.768 KHz crystal. Pin 10, on L3, should show this frequency on a scope while the crystal out pin, 11, should be at ground. Should these waveforms be absent, replace the filtering capacitors and retest. If the waveforms are still incorrect, replace the real time clock, L3, and retest.

A.3.2. SWITCHES

A.3.2.1. DISH ROTATION SWITCH TEST

This test procedure is used to test each connection in the four pole "DISH ROTATION" switch. All four lines are pulled up to five Volts through pull up resistors. The

line corresponding to the switch position is pulled down to ground.

Switch position number one corresponds to the right most position with the number increasing as the switch is turned to the left. Measure the voltage of all four peripheral interface pins. Using TABLE A-8 below, check to see that the pin corresponding to the current switch position is grounded while the remaining three pins are at five Volts.

This procedure must be repeated four times for each switch position. If one of the interface lines, not the corresponding switch position one, is not at five Volts, replace the pull up resistor and repeat the test.

SWITCH POSITION	PIA BIT	PIA PIN#
1	0	2
2	1	3
3	2	4
4	3	5

Table A-8: DISH ROTATION SWITCH POLES-PIA CONNECTIONS

A.3.2.2. LEARN/OPERATE SWITCH TEST

This test procedure is used to test the two position "LEARN/OPERATE" switch, more explicitly, the debouncing circuitry. This test is subdivided into three parts and must be performed twice, once for each switch position. The TABLE A-9, outlines the test components.

COMPONENT#	PIN#	COMPONENT#	PIN#
I1	1	H1	2
I1	15		

Table A-9: LEARN/OPERATE SWITCH-EC CONNECTIONS

First set the switch to the, lower, **LEARN** position. Measure the voltage at the first test point. If the voltage is not five Volts, there is a discontinuity between the switch and this point. If the problem exists at the edge connector, the wiring to the switch must be checked. If there is five Volts at the edge connector, the problem exists between the edge connector and the debouncing package.

If test point one is correct, test the second test point for five Volts. If this test fails, replace the hex debouncer, I1, and retest. If test point two is correct, test the third and final test point for five Volts. If this final test fails, a discontinuity exists between the hex debouncer and the peripheral interface adapter.

A.3.2.3. STANDBY SWITCH TEST

This test procedure is used to test the two position "STANDBY" switch, more explicitly, the debouncing circuitry. This test operates in the same manner as the previous **LEARN/OPERATE** switch test. The TABLE A-10 outlines the test components.

COMPONENT#	PIN#	COMPONENT#	PIN#
I1	1	H1	2
I1	15		

Table A-10: STANDBY SWITCH-EC CONNECTIONS

A.3.2.4. WAIT SWITCH TEST

This test procedure is used to test each connection in the four pole "WAIT" switch. This test is performed in the same manner as the **DISH** rotation test. Use TABLE A-11 to determine the appropriate pole - **PIA** pin combinations.

SWITCH POSITION	PIA BIT	PIA PIN#
1	0	6
2	1	7
3	2	8
4	3	9

Table A-11: WAIT SWITCH POLE-PIA CONNECTIONS

A.4. DIAGNOSTIC DISPLAY VALUES

TEST #	TEST ID
01-02	PTM TEST 1-5
06-14	PIA TEST 1-9
15	LED TEST
16	LEARN SWITCH TEST
17	STANDBY SWITCH TEST
18	REPEAT SWITCH TEST
19	DISH ROTATION SWITCH TEST
20	WAIT SWITCH TEST
21	SPEED CONTROL SWITCH TEST
22	DIAGNOSTIC DISPLAY TEST
23	STEPPER MOTOR MOVEMENT TEST 6
24	MANUAL SWITCH TEST
25	REAL TIME CLOCK TEST
27	VOICE RECOGNITION
28-35	RAM TEST 1-8
36-37	ACIA TEST 1-2
38-42	STEPPER MOTOR DIRECTION TEST 1-5
43-47	STEPPER MOTOR PULSE DETECTION TEST 1-5
48-52	STEPPER MOTOR MOVEMENT TEST 1-5
53	STEPPER MOTOR RETURN TEST.
54	ROBOZ MONITOR.

B. OPERATOR'S MANUAL

Operation of the robotic manipulator can be subdivided into two modes, "operation" and "instruction". The instruction phase is to program the manipulator for automatic feeding operation. Likewise, the operating mode can be divided into "manual" and "automatic" operation. Finally, "manual" operation may be performed by two different interfaces, "keypad" and "audio input".

B.1. AUTOMATIC FEEDING OPERATION

The "automatic feeding" mode is dedicated to a single user. After the manipulator has been taught the necessary movements to bring the feeding utensil to the patient's mouth, the system is ready for "automatic feeding" operation. To bring food to the user's mouth, simply depress the repeat switch. This will cause the manipulator to execute the programmed moves and return to "HOME BASE" to await the next command.

To program the robotic arm the following switch settings must be met. First, set the "MANUAL/AUTOMATIC" switch to the automatic position noting that the upper green ("AUTOMATIC") LED is now lit. Move the "LEARN/OPERATE" switch to the "OPERATE" position which turns the yellow ("OPERATE") LED on. The manipulator will now automatically move to the "HOME BASE" position, in preparation for the teaching process.

To begin programming, move the "LEARN/OPERATE" to

the "LEARN" position. Note that this operation turns the lower red ("LEARN") **LED** on. Now begin programming by moving the arm through the desired motions. When finished, move the "LEARN/OPERATE" switch to the "OPERATE" setting and wait for the "OPERATE" **LED** to turn on. The instruction phase is now complete.

B.2. MANUAL OPERATION

"Manual" control of the manipulator may be achieved by two different methods. Keypad control is provided when the "KEYPAD/VOICE" switch is in the lower position. The corresponding green ("KEYPAD") **LED** should be lit to indicate keypad control. For the second case, "audio input", the switch must be in the upper ("VOICE") position with the corresponding red **LED** lit. For both cases, movement is identical only the method of receiving commands differs.

B.2.1. KEYPAD OPERATION

All of the basic movement commands are found on the 16 button keypad. Figure B-1 illustrates the operations associated with each key. For efficient operation, note the structure of the function layout.

The keys corresponding to up, down, left, right and stop are oriented accordingly. The stop button is in the center while the right function is to the right of stop, the up function is above stop, etc.. The functions for forwards, backwards and home have also been strategically

placed.

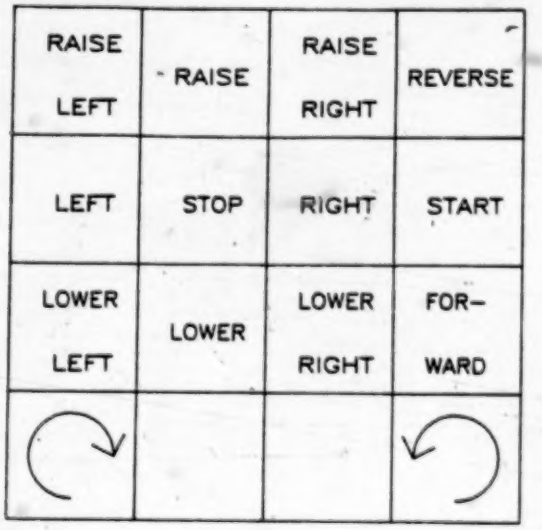


Fig. B-1: KEYPAD LAYOUT

B.2.2. AUDIO OPERATION

"Audio" operation is capable of performing the same set of commands as "keypad" operation. The instructions are accessed using a cyclic serial menu method. Upon entry, the last instruction is executed until an initialization prompt is sent via the microphone.

After the initialization prompt is received, instruction execution is terminated and the audio menu is entered. Each command is heard with an approximate two second delay in between phrases. When the desired command is heard, an acknowledge signal must be sent. The audio

menu is terminated and instruction execution begins.

B.3. ADDITIONAL FEATURES

There are four functions provided for configuring the system to a patient's needs. Two rotary switches manage the manipulator movements. The first, manipulator speed control, governs the speed during both manual and programmed operation. The wait switch is used during programmed operation only and sets the delay between movement completion and the return to home base. This correpsonds to the length of time which the eating utensil will remain at the patient's mouth.

The two remaining features, dish rotation and volume, are not required by both modes of operation. The dish rotation switch is used during programmed operation and sets the degree of dish rotation. The loudness control is used during manual operation only, and regulates the volume.

B.4. SWITCH SETTING SUMMARY

OPERATION	KEYPAD/VOICE	MANUAL/AUTOMATIC	LEARN/OPERATE
Keypad control	Keypad	Manual	Don't care
Voice control	Voice	Manual	Operate
Feed mode	Don't care	Automatic	Operate
Feed learn	Don't care	Automatic	Learn

B.5. KEYPAD SUMMARY

ROW	COL	OPERATION
1	1	Raise, left
1	2	Raise
1	3	Raise, right
1	4	Reverse
2	1	Left
2	2	Stop
2	3	Right
2	4	Start
3	1	Lower, left
3	2	Lower
3	3	Lower, right
3	4	Forward
4	1	Restore
4	2	
4	3	
4	4	Rotate

C. CIRCUIT DIAGRAMS

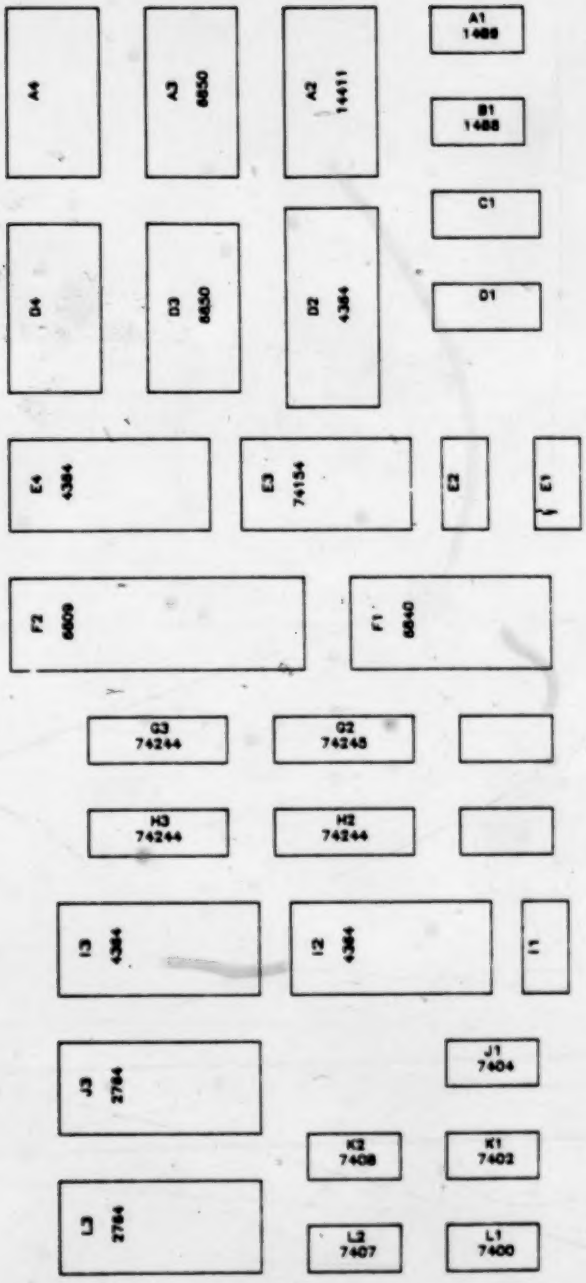


Fig. C-1: BOARD A, BOARD LAYOUT

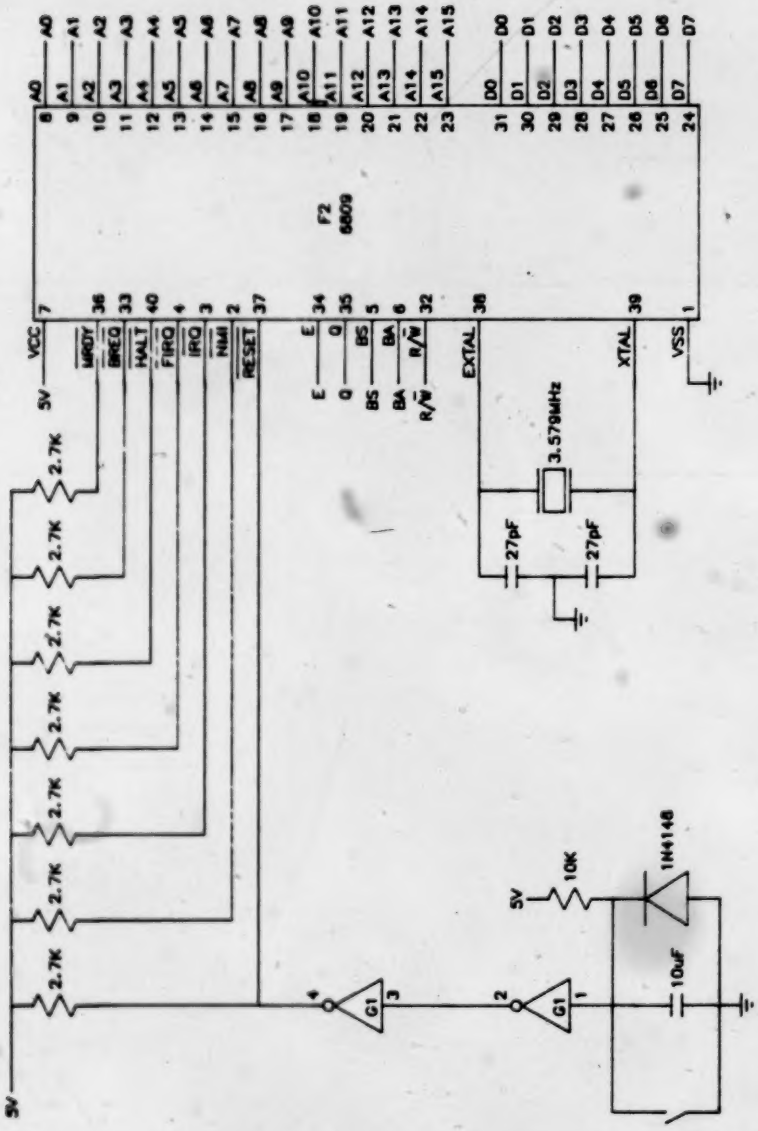


Fig. C-2: BOARD A, MICROPROCESSOR AND RESET

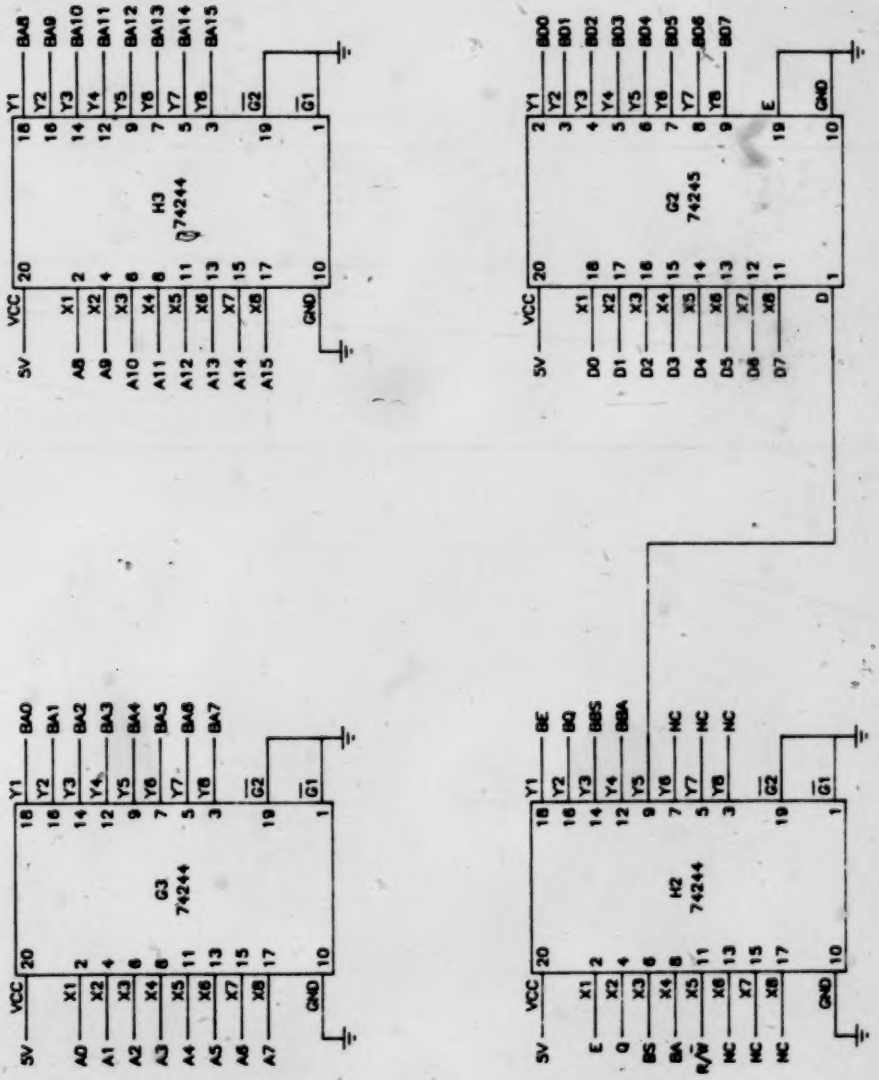


Fig. C-3: BOARD A, BUFFERS

SIGNAL DESIGNATION	ADDRESSING RANGE	UPPER BOUND
SEL0	0000–0FFF	4K
SEL1	1000–1FFF	8K
SEL2	2000–2FFF	12K
SEL3	3000–3FFF	16K
SEL4	4000–4FFF	20K
SEL5	5000–5FFF	24K
SEL6	6000–6FFF	28K
SEL7	7000–7FFF	32K
SEL8	8000–8FFF	36K
SEL9	9000–9FFF	40K
SELA	A000–AFFF	44K
SELB	B000–BFFF	48K
SELC	C000–CFFF	52K
SELD	D000–DFFF	56K
SELE	E000–EFFF	60K
SELF	F000–FFFF	64K

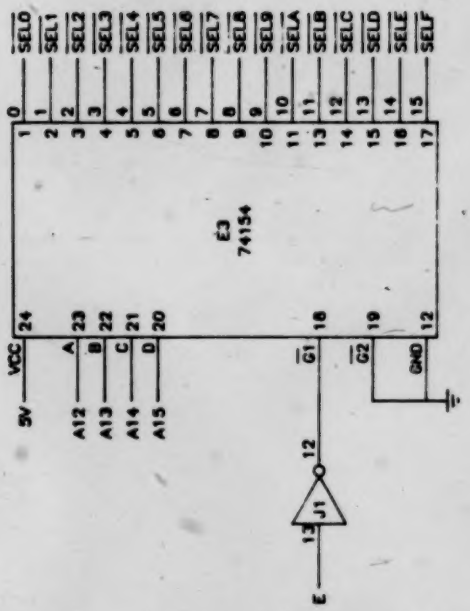


Fig. C-4: BOARD A, ADDRESS DECODING

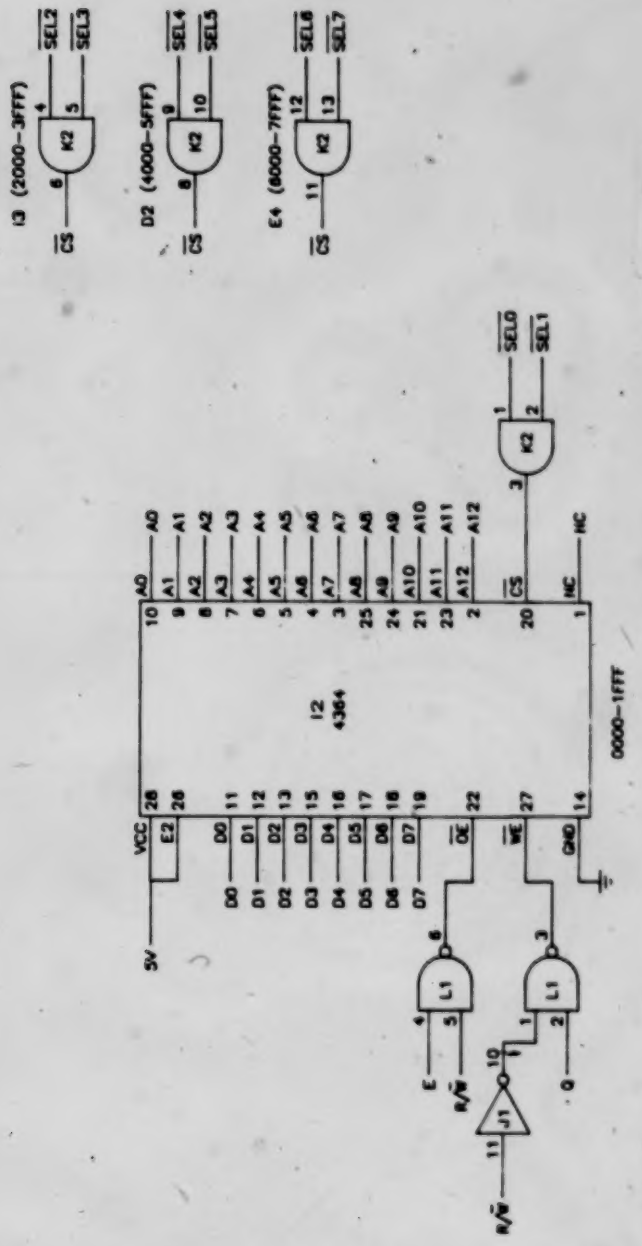


Fig. C-5: BOARD A, USER RAM

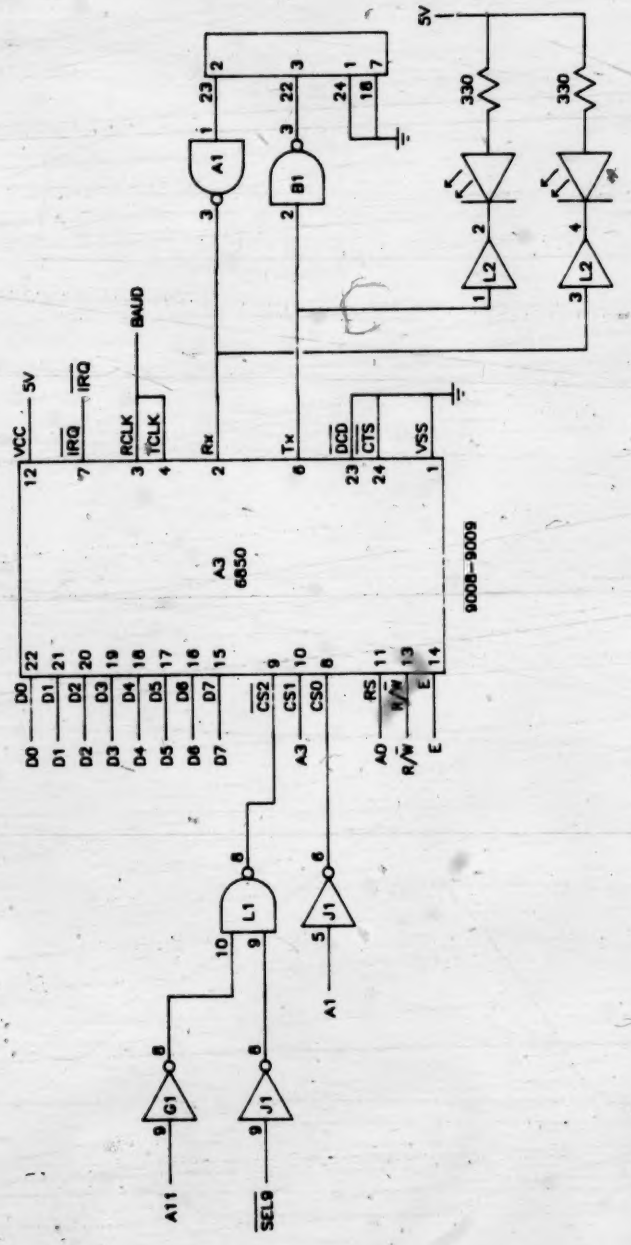


Fig. C-6: BOARD A, HOST ACIA

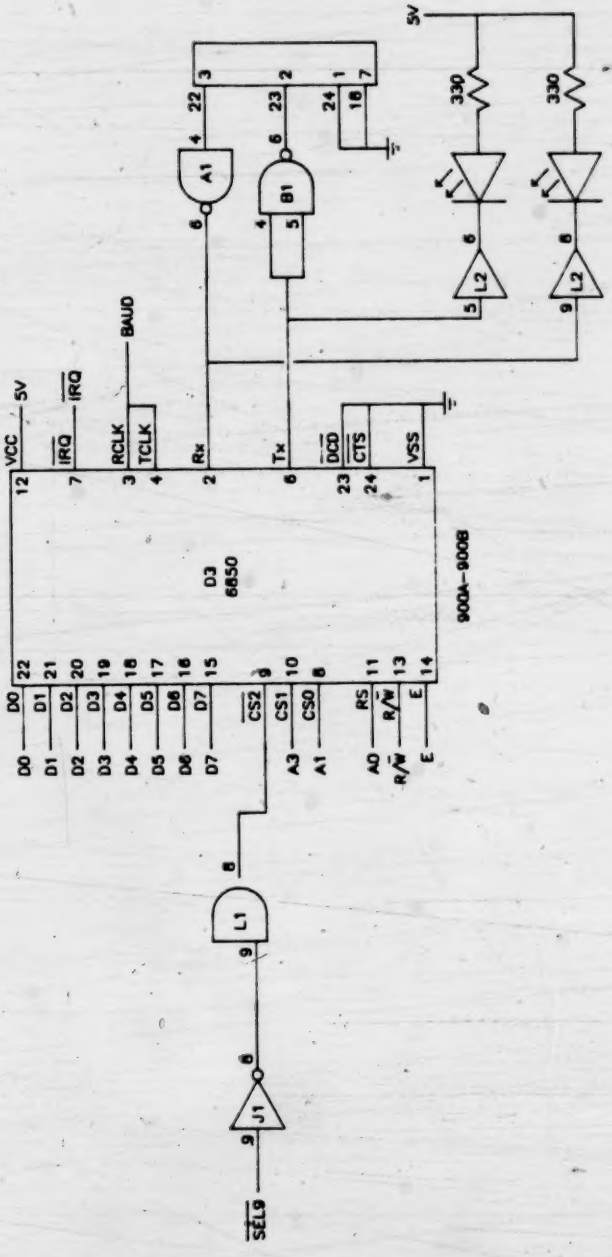


Fig. C-7: BOARD A, TERMINAL ACIA

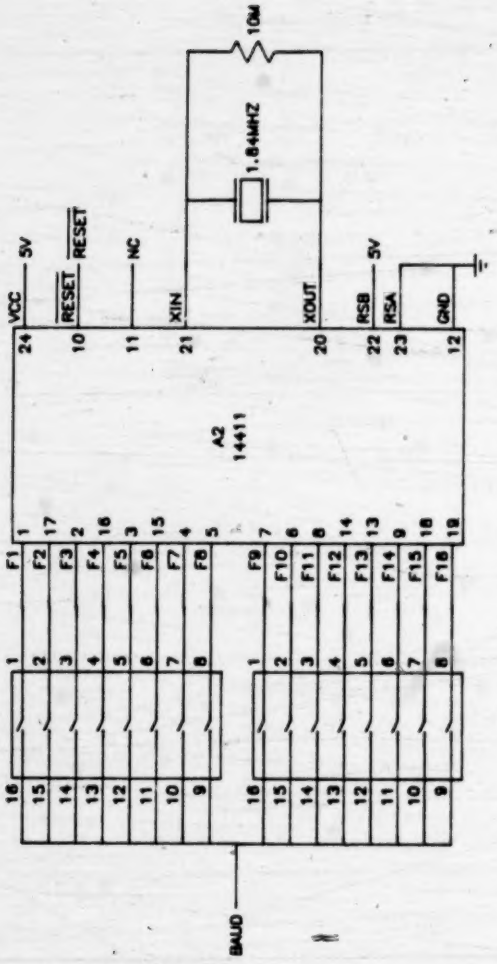


Fig. C-8: BOARD A, SELECTABLE BAUD RATE GENERATOR

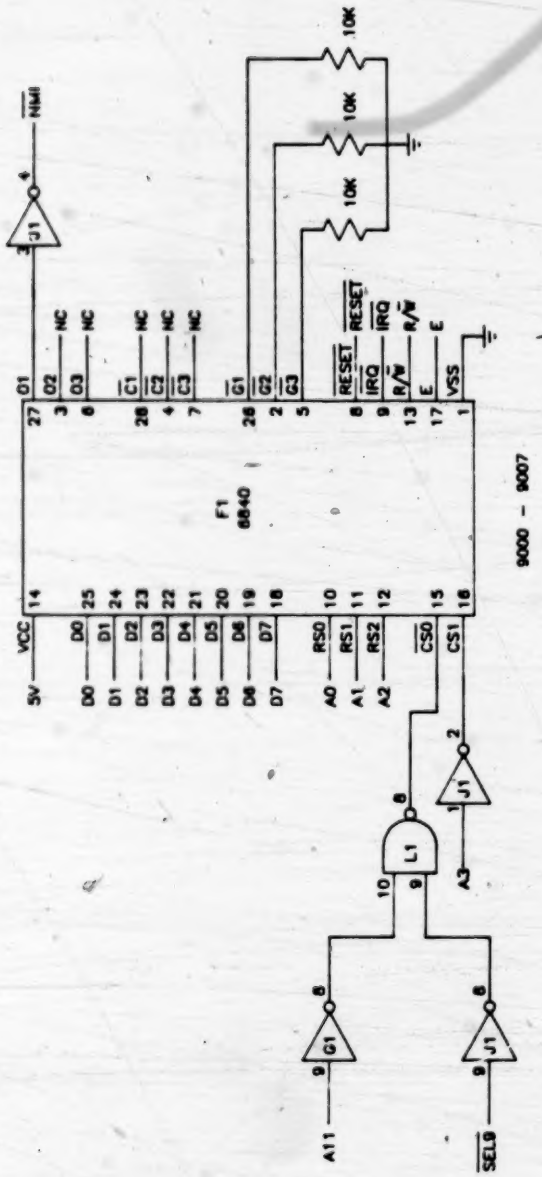


Fig. C-9: BOARD A, PTM 3

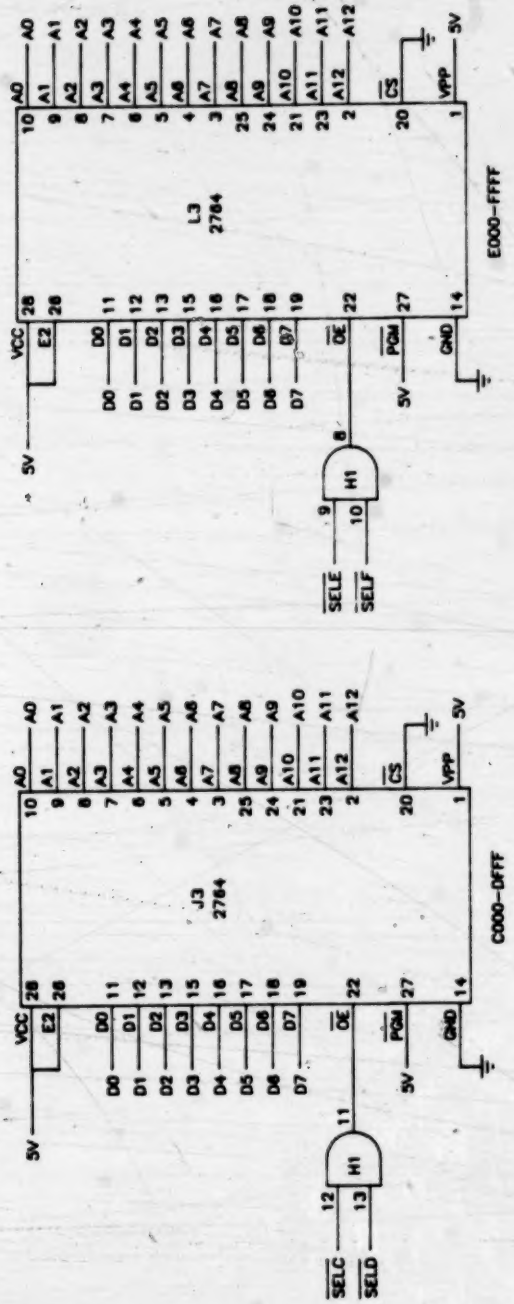


Fig. C-10: BOARD A, USER ROM

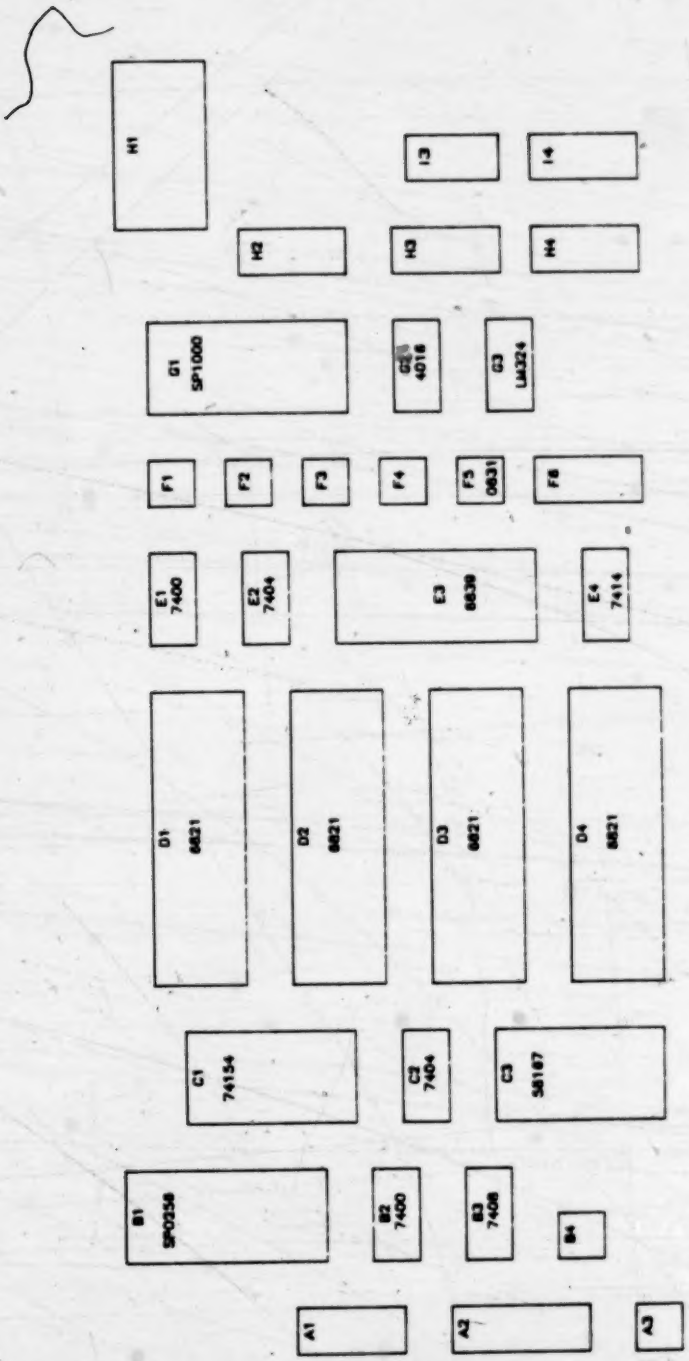


Fig. C-11: BOARD B, BOARD LAYOUT

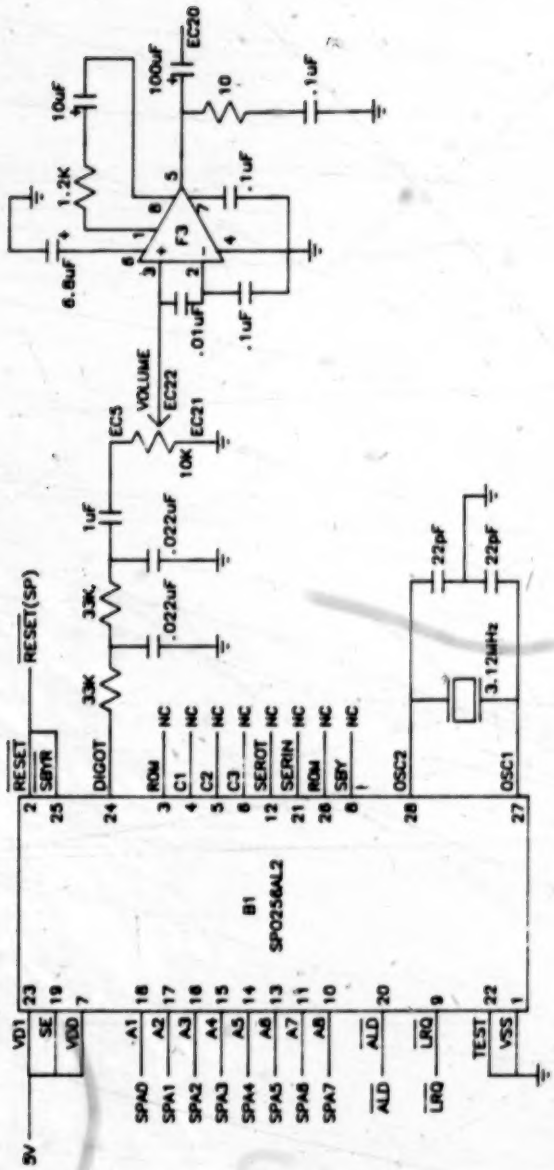


Fig. C-12: BOARD B, SPEECH PROCESSOR AND AUDIO AMPLIFIER

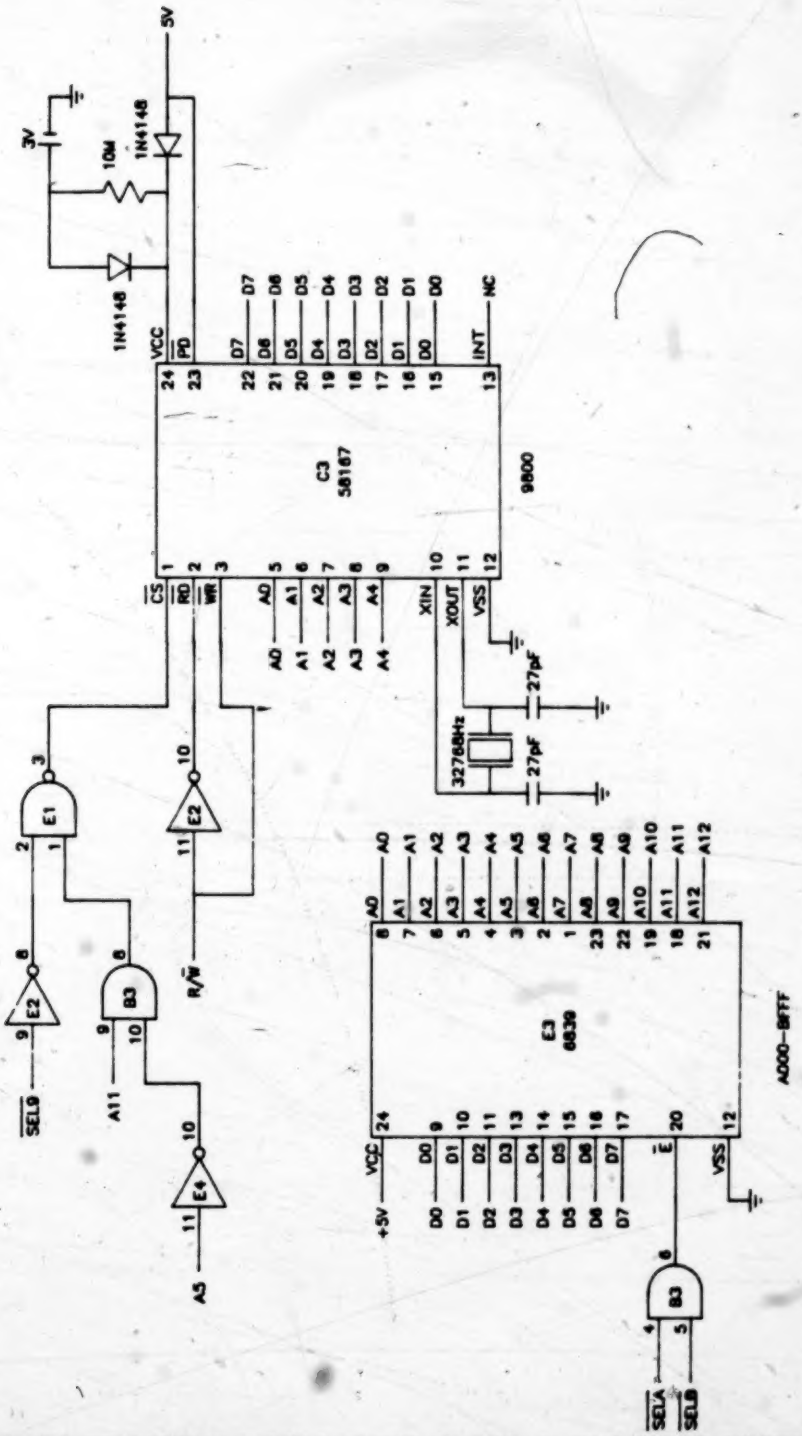


Fig. C-13: BOARD B, REAL TIME CLOCK & FLOATING POINT ROM

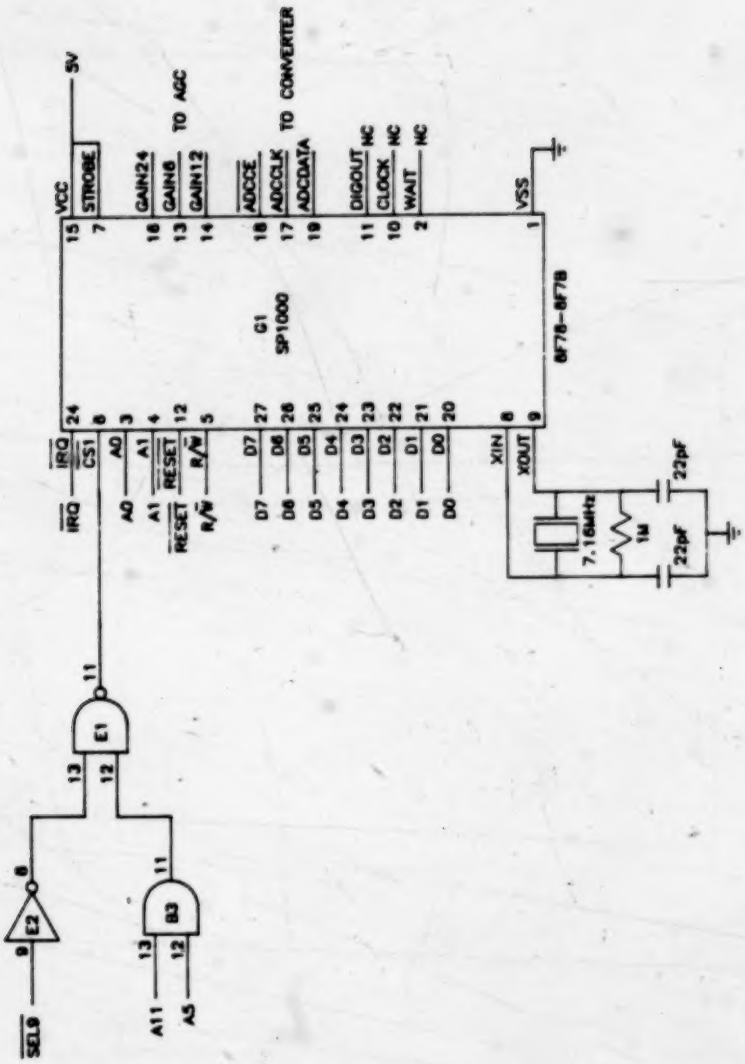


Fig. C-14: BOARD B, AUDIO INPUT

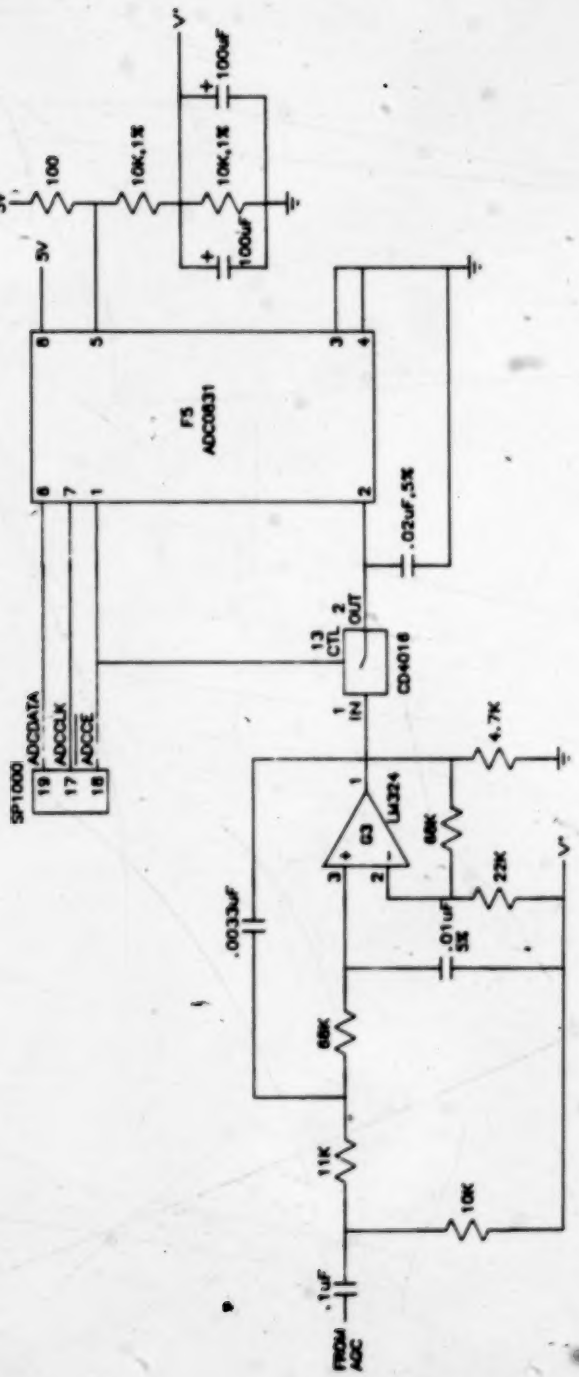


Fig. C-15: BOARD B, SAMPLE AND HOLD + ANTIALIASING FILTER

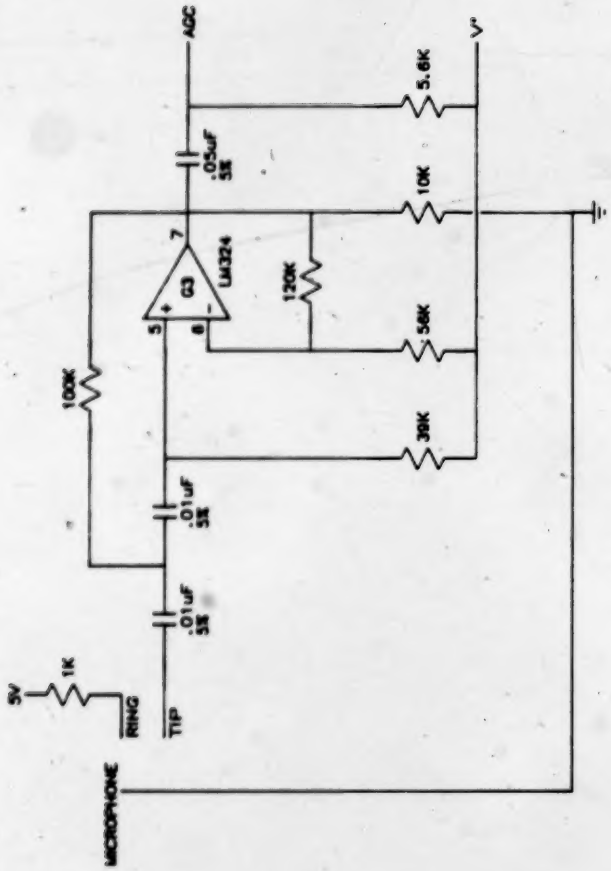


Fig. C-16: BOARD B, HIGH PASS FILTER FOR MICROPHONE

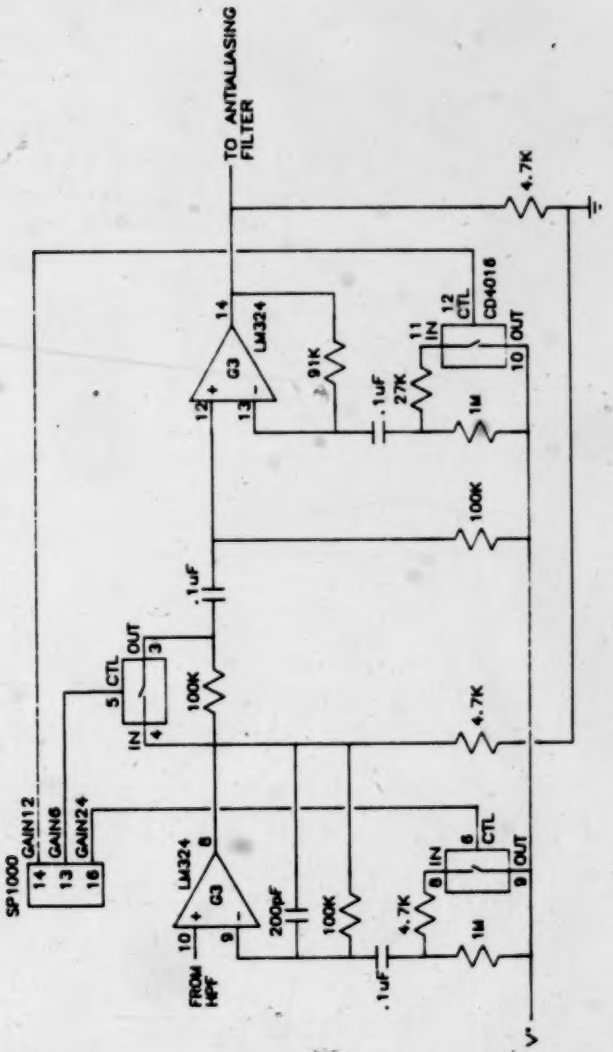


Fig. C-17: BOARD B, PROGRAMMABLE AGC

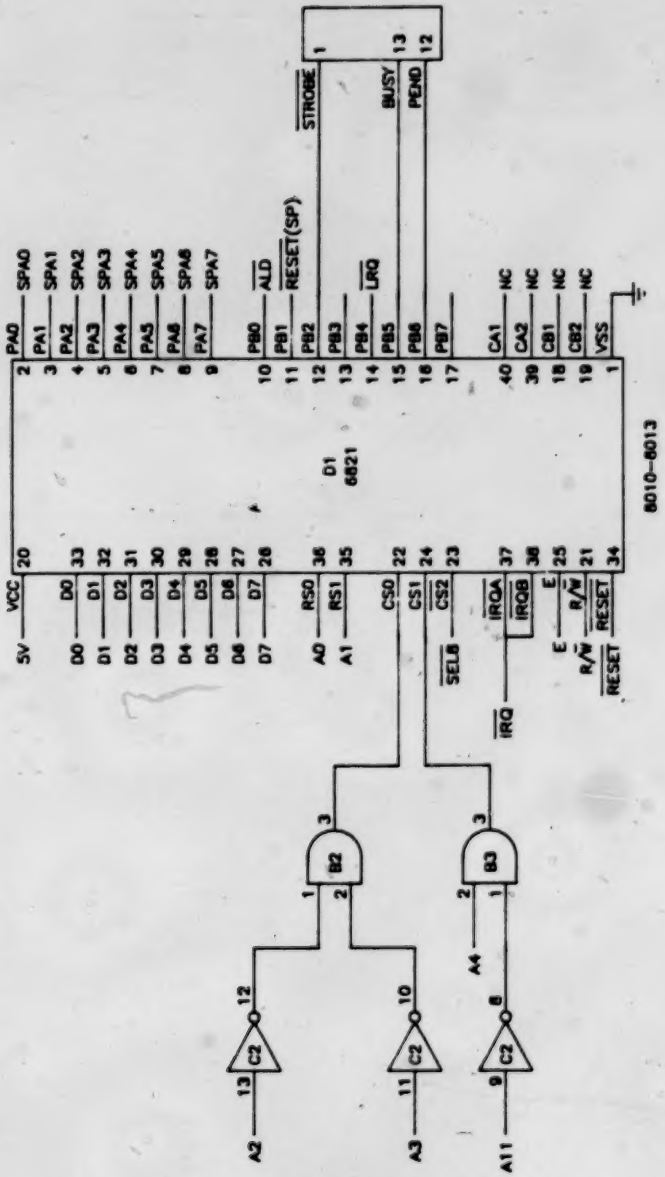


Fig. C-18: BOARD B, PIA 5

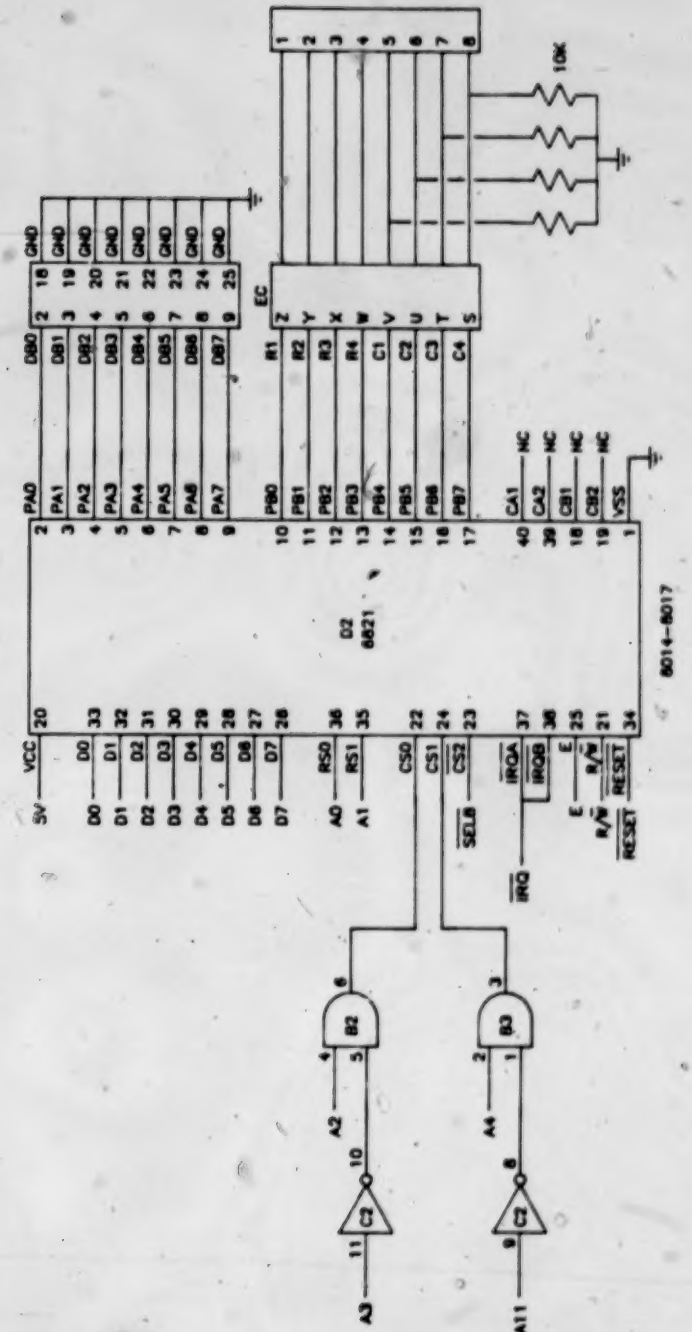


Fig. C-19: BOARD B, PIA 6

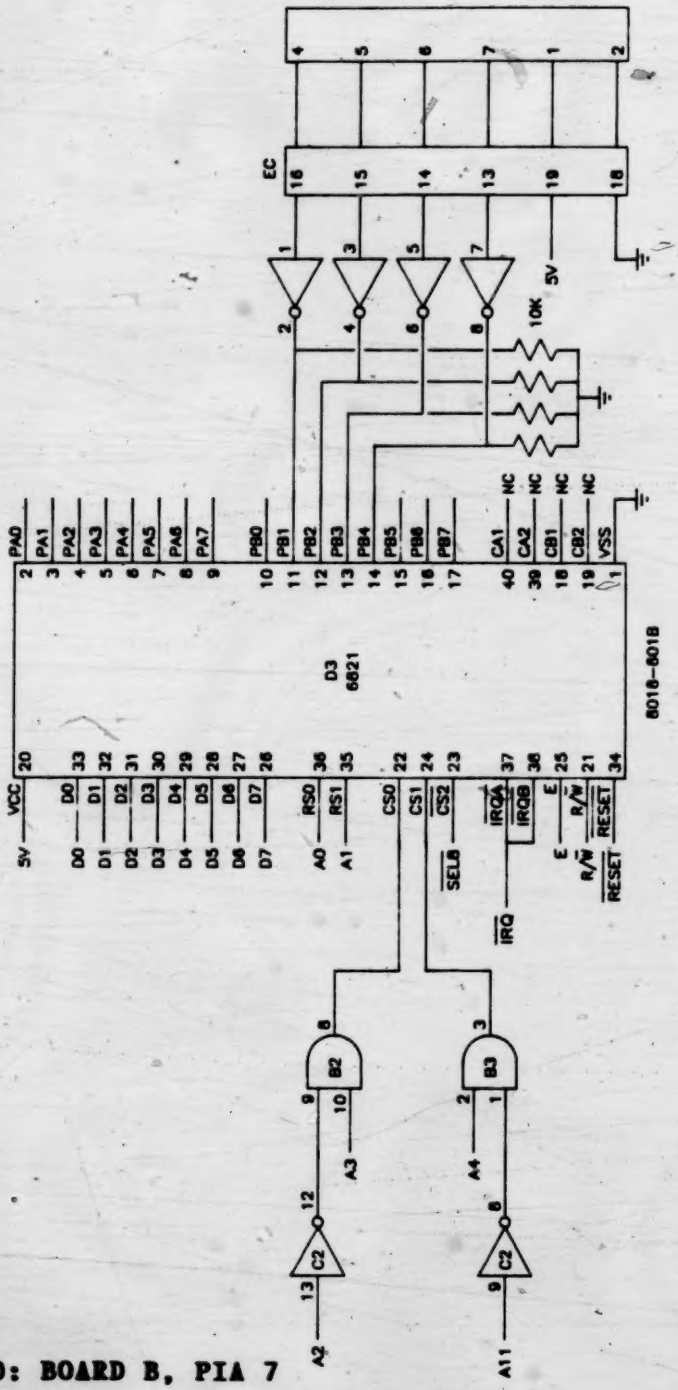


Fig. C-20: BOARD B, PIA 7

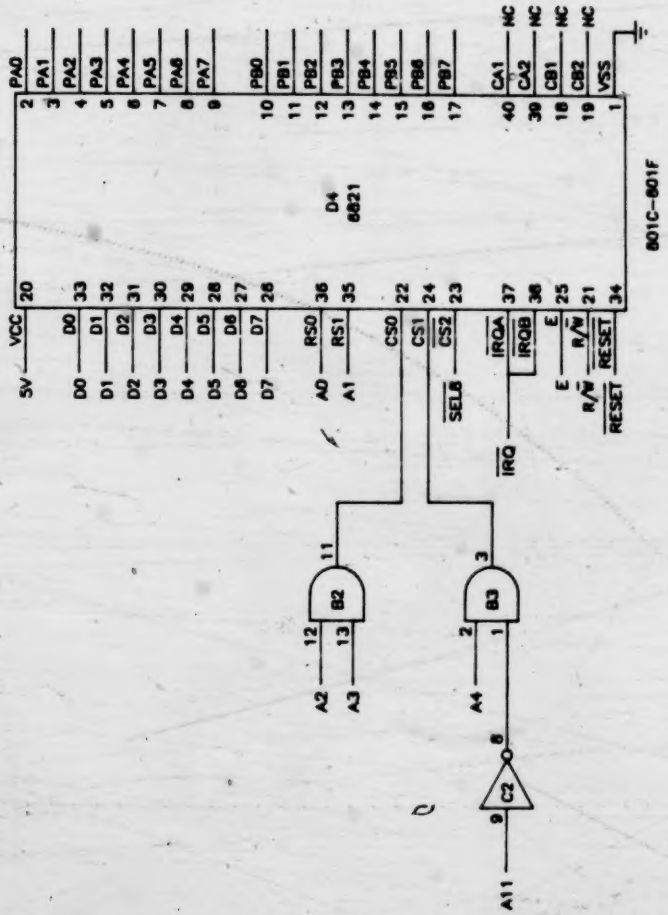


Fig. C-21: BOARD B, PIA 8

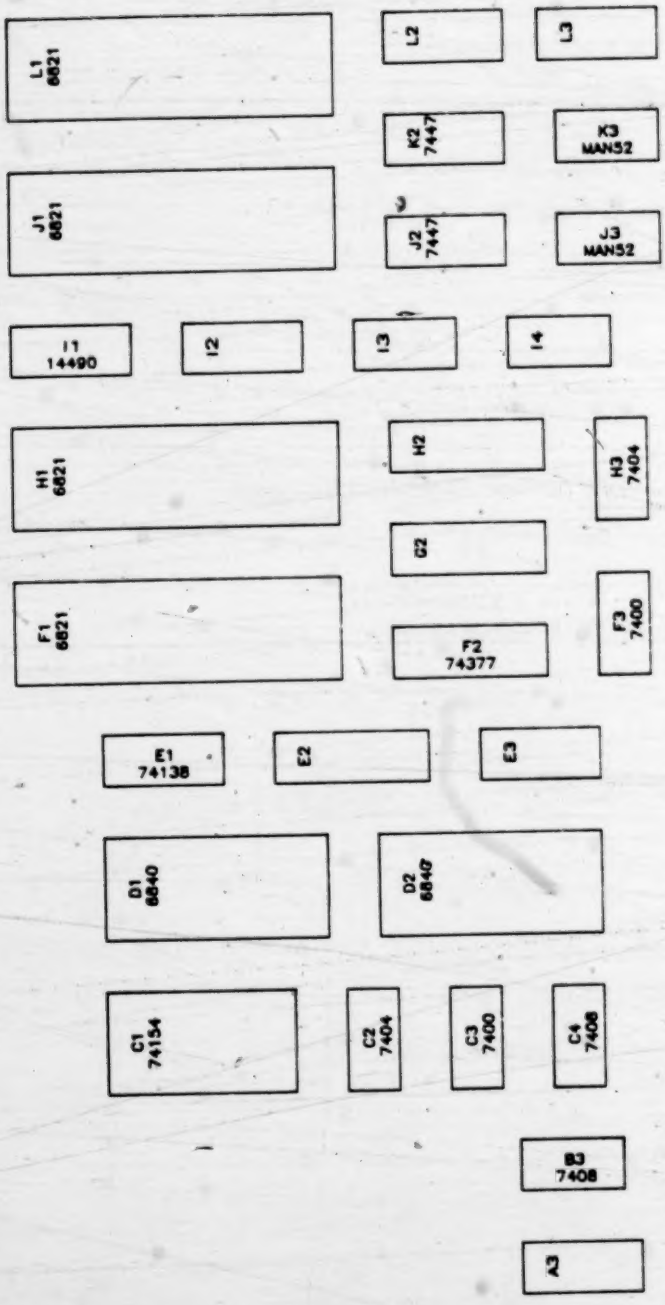


Fig. C-22: BOARD C, BOARD LAYOUT

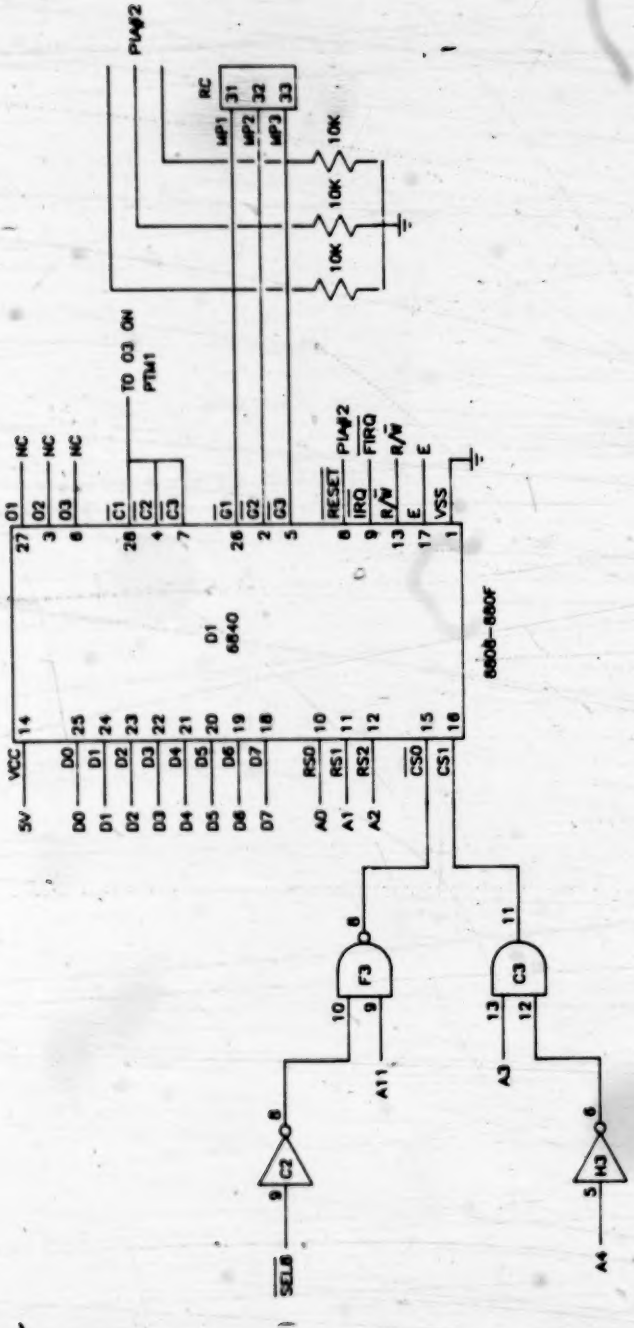


Fig. C-23: BOARD C, PTM 2

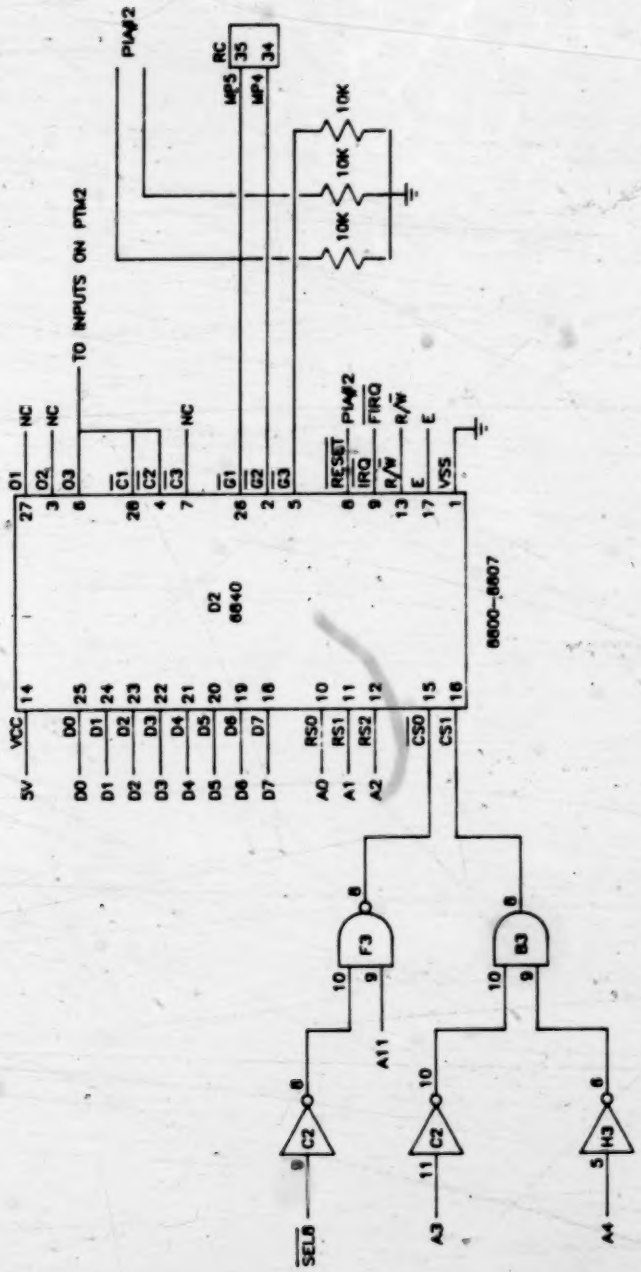


Fig. C-24: BOARD C, PTM 1

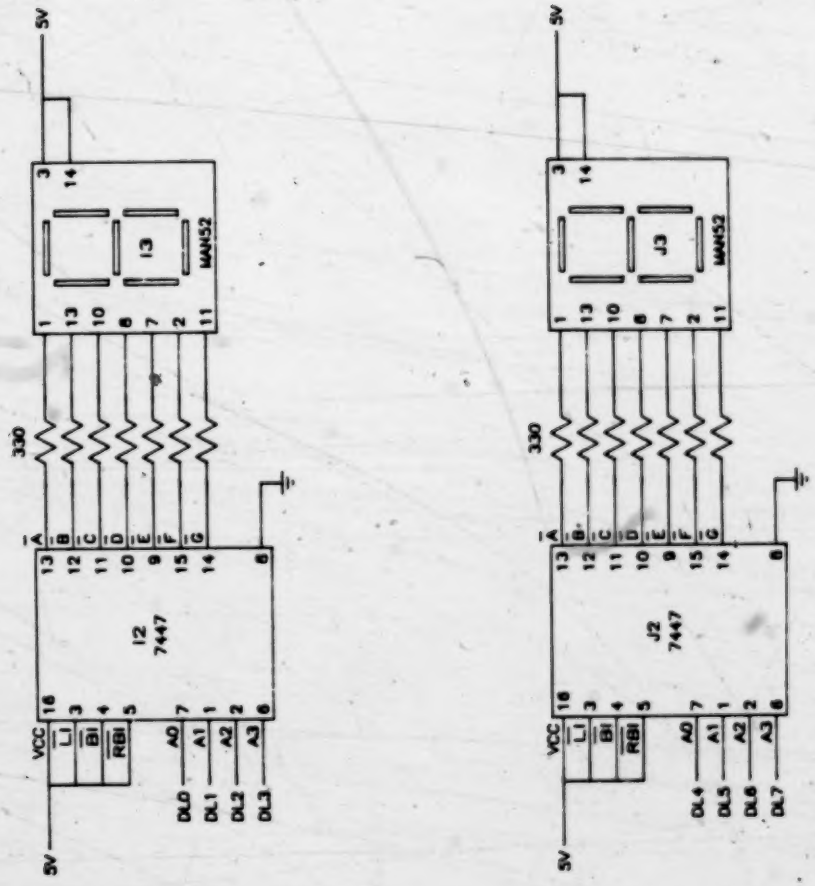


Fig. C-25: BOARD C, DIAGNOSTIC DISPLAY

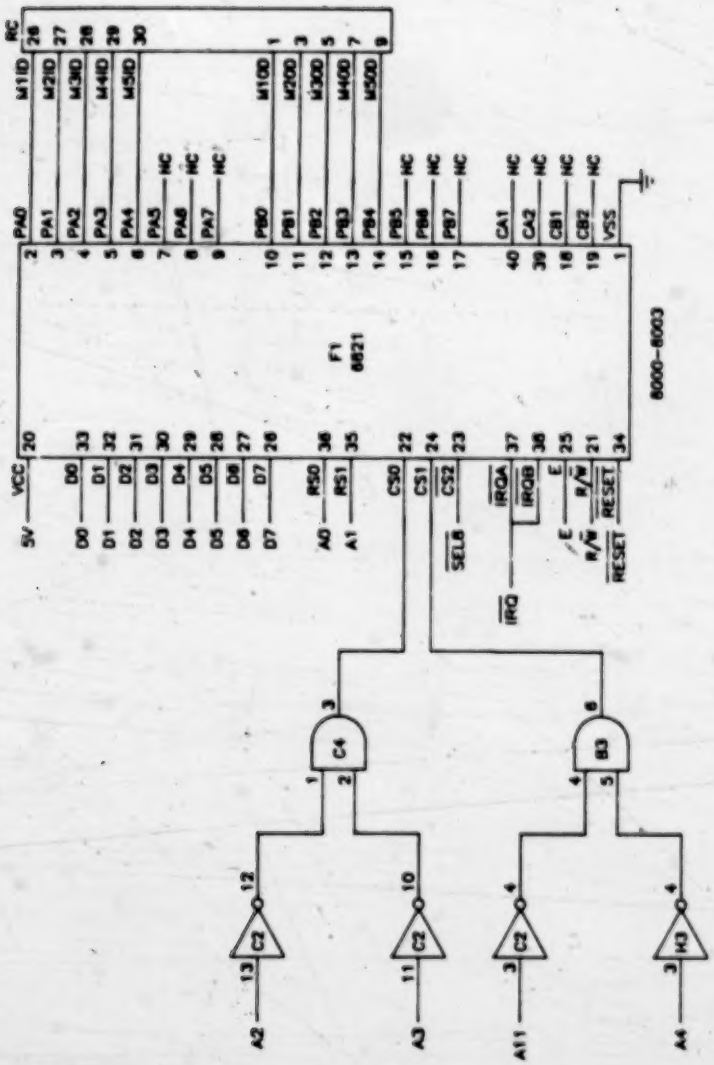


Fig. C-26: BOARD C, PIA 1

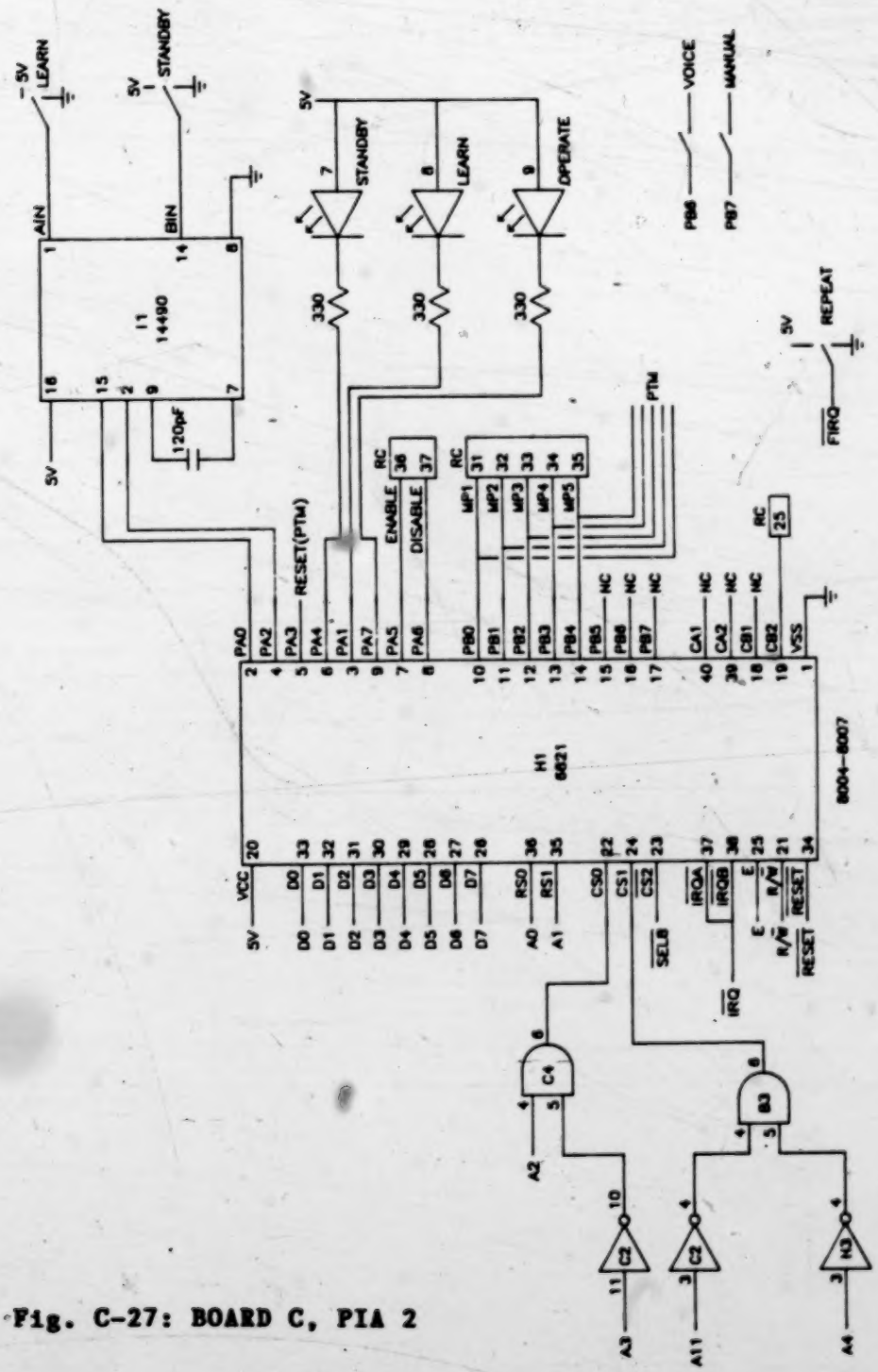


Fig. C-27: BOARD C, PIA 2

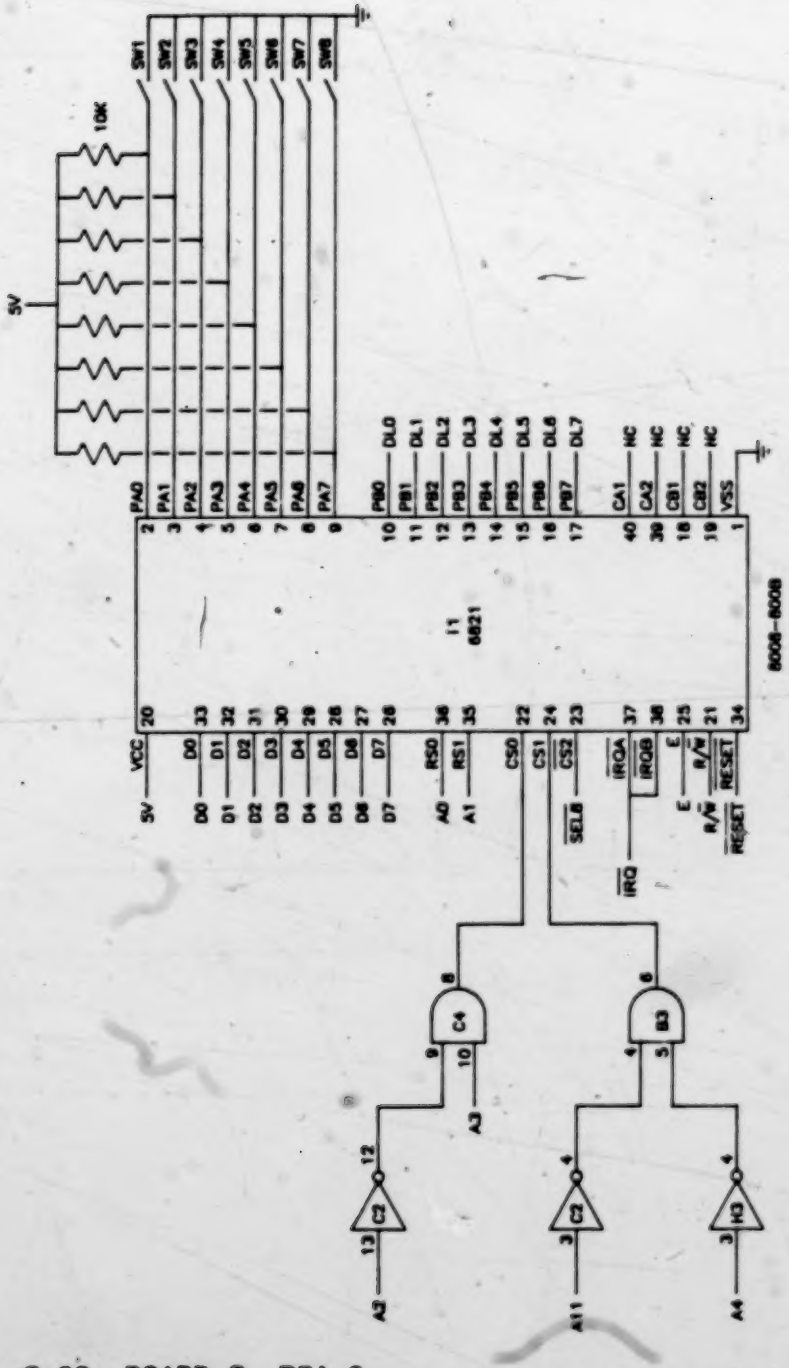


Fig. C-28: BOARD C, PIA 3

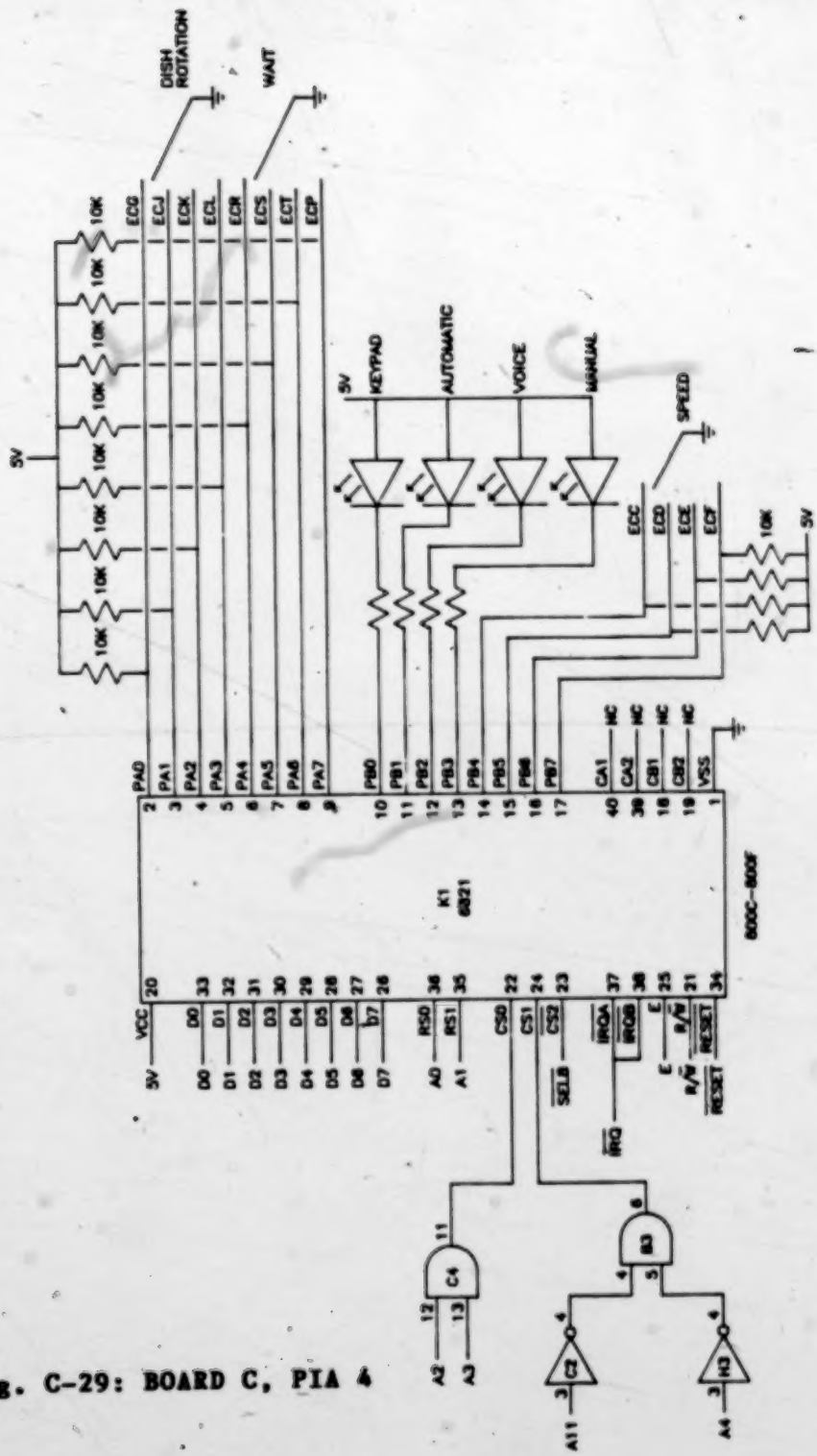


Fig. C-29: BOARD C, PIA 4

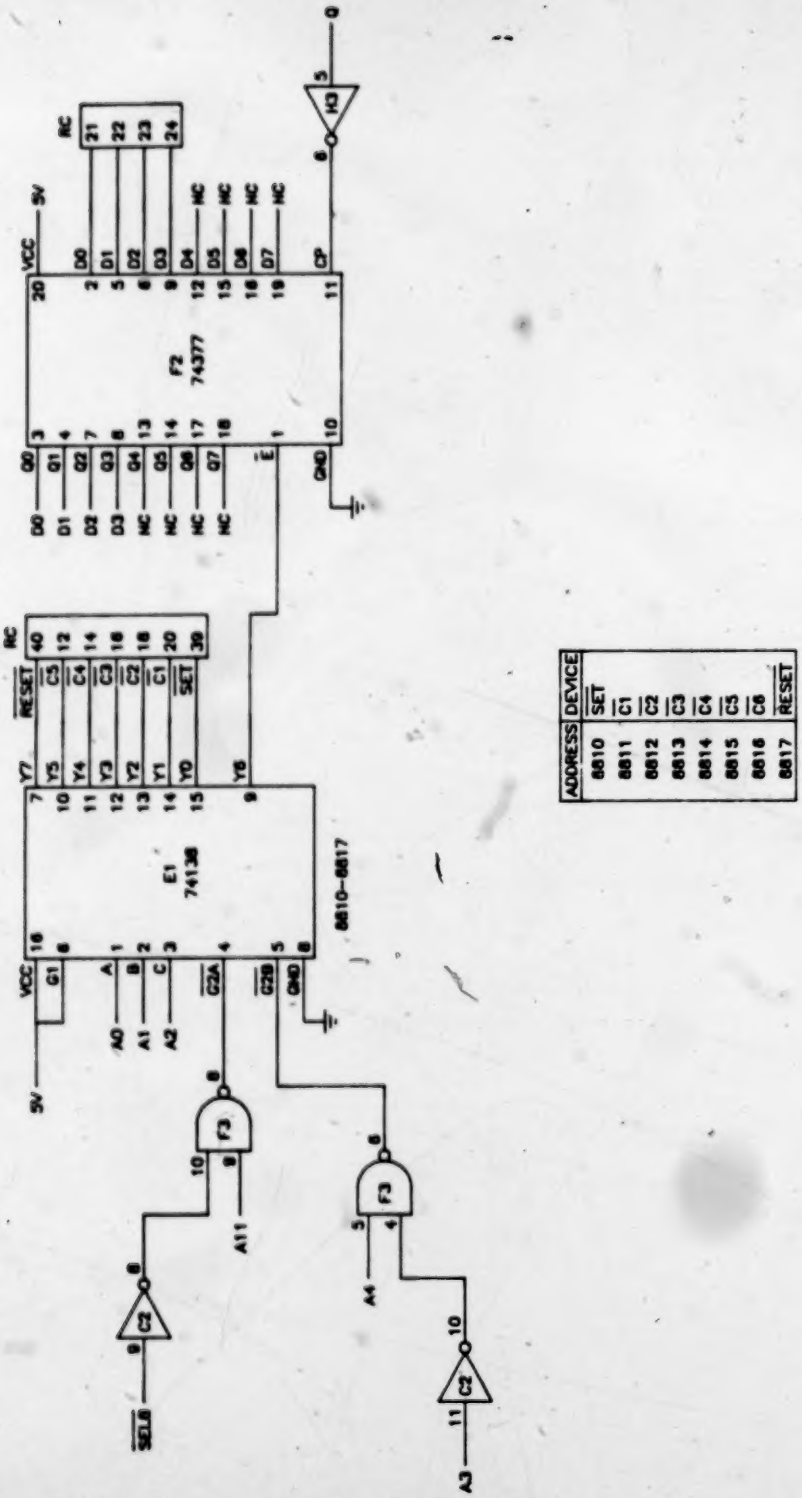


Fig. C-30: BOARD C, STEPPER MOTOR SELECT

BOARD	ADDRESS	DESCRIPTION
A	F800–FFFF	MONITOR ROM
A	C000–F7FF	USER ROM
B	A000–BFFF	FLOATING POINT ROM
B	9800–981F	REAL TIME CLOCK
A	900A–900B	HOST ACIA
A	9008–9009	TERMINAL ACIA
A	9000–9007	PTM 3
B	8F78–8F7B	SPEECH RECOGNITION
C	8810–8817	PTM 2
C	8808–880F	PTM 1
C	8800–8807	PTM 2
B	801C–801F	PIA 8
B	8018–801B	PIA 7
B	8014–8017	PIA 6
B	8010–8013	PIA 5
C	800C–800F	PIA 4
C	8008–800B	PIA 3
C	8004–8007	PIA 2
C	8000–8003	PIA 1
A	0000–7FFF	USER RAM

Fig. C-31: MEMORY MAP

PIN	SIGNAL	PIN	SIGNAL
1	M10D	40	$\overline{\text{RESET}}$
2		39	$\overline{\text{SET}}$
3	M20D	38	
4		37	DISABLE
5	M30D	36	ENABLE
6		35	MP5
7	M40D	34	MP4
8		33	MP3
9	M50D	32	MP2
10		31	MP1
11		30	M5ID
12	$\overline{\text{C5}}$	29	M4ID
13		28	M3ID
14	$\overline{\text{C4}}$	27	M2ID
15		26	M1ID
16	$\overline{\text{C3}}$	25	CB2
17		24	M6P4
18	$\overline{\text{C2}}$	23	M6P3
19		22	M6P2
20	$\overline{\text{C1}}$	21	M6P1

PIN	SIGNAL	PIN	SIGNAL
1	A0	34	D0
2	A1	33	D1
3	A2	32	D2
4	A3	31	D3
5	A4	30	D4
6	A5	29	D5
7	A6	28	D6
8	A7	27	D7
9	A8	26	E
10	A9	25	Q
11	A10	24	BS
12	A11	23	BA
13	A12	22	R/$\overline{\text{W}}$
14	A13	21	$\overline{\text{IRQ}}$
15	A14	20	$\overline{\text{RESET}}$
16	A15	19	$\overline{\text{NMI}}$
17		18	$\overline{\text{FIRQ}}$

Fig. C-32: RIBBON CABLE CONNECTORS

DC39	EDGE	SIGNAL	MOTOR	COLOR
1	1	M1COM	1	ORANGE
2	2	M1YP1	1	ORANGE
3	3	M1GP1	1	ORANGE
4	4	M1YP2	1	ORANGE
5	5	M1GP2	1	ORANGE
6	6	M2COM	2	BLUE
7	7	M2YP1	2	BLUE
8	8	M2GP1	2	BLUE
9	9	M2YP2	2	BLUE
10	10	M2GP2	2	BLUE
11	11	M3COM	3	YELLOW
12	12	M3YP1	3	YELLOW
13	13	M3GP1	3	YELLOW
14	14	M3YP2	3	YELLOW
15	15	M3GP2	3	YELLOW
28	A	M4COM	4	WHITE
29	B	M4YP1	4	WHITE
30	C	M4GP1	4	WHITE
31	D	M4YP2	4	WHITE
32	E	M4GP2	4	WHITE
23	F	M5COM	5	GREEN
24	H	M5YP1	5	GREEN
25	J	M5GP1	5	GREEN
26	K	M5YP2	5	GREEN
27	L	M5GP2	5	GREEN
18	16	M6COM	6	GREY
19	17	M6YP1	6	GREY
20	18	M6GP1	6	GREY
21	19	M6YP2	6	GREY
22	20	M6GP2	6	GREY

Fig. C-33: D CONNECTOR FOR STEPPER MOTORS